Report No. CG-D-10-89

AD-A211 786

STATISTICAL MODELS FOR THE OPTIMAL ESTIMATION OF OCEANIC FIELDS

Prepared by:

L. CHARLES SUN*
ARTHUR A. ALLEN
AND
CLARE B. BILLING, JR.

U.S. COAST GUARD RESEARCH AND DEVELOPMENT CENTER
AVERY POINT
GROTON, CONNECTICUT 06340-6096

INTERIM REPORT
JUNE 1989

* Now at: Marine Sciences Building, Rm. 424
University of Hawaii
1000 Pope Road
Honolulu, Hawaii 96822

This document is available to the U.S. public through the
National Technical Information Service, Springfield, Virginia 22161

Prepared for:

U.S. Department Of Transportation
United States Coast Guard
Office of Engineering, Logistics, and Development
Washington, DC 20593-0001

NOTICE

SAMUEL F. POWEL, III
Technical Director
U.S. Coast Guard Research and Development Center
Avery Point, Groton, Connecticut 06340-6096

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
CG-D-10-89		

4. Title and Subtitle STATISTICAL MODELS FOR THE OPTIMAL ESTIMATION OF OCEANIC FIELDS	5. Report Date JUNE 1989 6. Performing Organization Code
7. Author(s) L. Charles Sun, Arthur A. Allen, and Clare B. Billing, Jr.	8. Performing Organization Report No. R&DC 10/87
9. Performing Organization Name and Address U.S. Coast Guard Research and Development Center Avery Point Groton, Connecticut 06340-6096	10. Work Unit No. (TRAIS) 11. Contract or Grant No.
12. Sponsoring Agency Name and Address Department of Transportation U.S. Coast Guard Office of Engineering, Logistics, and Development Washington, D.C. 20593-0001	13. Type of Report and Period Covered INTERIM 14. Sponsoring Agency Code

15. Supplementary Notes

This report is the twenty-fourth in a series which documents the Improvement in Probability of Detection in Search and Rescue (POD/SAR) Project at the U.S. Coast Guard R&D Center.

16. Abstract

The Oceanography Branch is conducting an ongoing investigation into the drift and leeway of survivors and survivor crafts as part of the Improvement of Probability of Detection in Search and Rescue Project. Part of this effort uses freely drifting buoys that transmit their positions to a shore/ship-based receiver to provide an estimate of the surface current field. The data sets are in the form of drift tracks from each of the drifters. To analyze the irregular spaced drifter tracks, objective analysis techniques were applied to produce optimal estimations of surface current fields on a regularly spaced grid. The transformation of a data set onto a regularly spaced grid allows for the use of many other standard computer analysis programs. A computer program using this technique has previously been successfully applied to large oceanographic data sets of many drifter tracks. The objective analysis program was converted for use on Hewlett Packard microcomputers and applied to a much more limited data set from six drifters. The results being that objective analysis can effectively work with small data sets.

The effect of very limited data sets on the outcome of the estimated fields was checked by removing drifters, one at a time, from the data set of six drifters. The fields generated by the reduced number of drifters were then compared to the results with the "best estimate" field determined by all six drifters. The results from this exercise are that the placement of the buoys or drifters relative to the flow field features will greatly influence how well the total flow field is estimated. A few well placed buoys will provide better information than many poorly placed buoys. Therefore, the use of remotely sensed data will aid in the determination of the optimal placement of buoys.

17. Key Words	18. Distribution Statement
Search and Rescue, drifting buoys, current fields, objective analysis	Document is available to the U.S. public through the National Technical Information Service, Springfield, Virginia 22161

19. Security Classif. (of this report)	20. SECURITY CLASSIF. (of this page)	21. No. of Pages	22. Price
UNCLASSIFIED	UNCLASSIFIED		

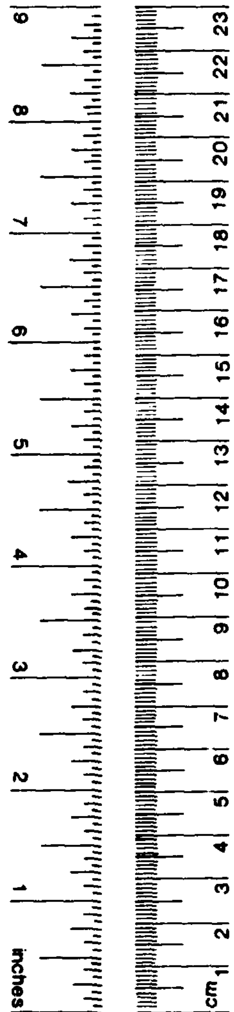
Form DOT F 1700.7 (8/72) Reproduction of form and completed page is authorized

METRIC CONVERSION FACTORS

Approximate Conversions to Metric Measures

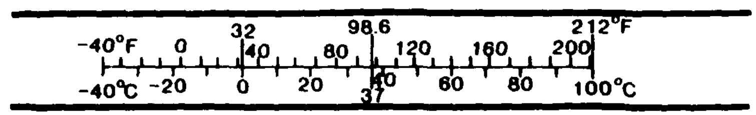
Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
in	inches	*2.5	centimeters	cm
ft	feet	30	centimeters	cm
yd	yards	0.9	meters	m
mi	miles	1.6	kilometers	km
		AREA		
in^2	square inches	6.5	square centimeters	cm^2
ft^2	square feet	0.09	square meters	m^2
yd^2	square yards	0.8	square meters	m^2
mi^2	square miles	2.6	square kilometers	km^2
	acres	0.4	hectares	ha
		MASS (WEIGHT)		
oz	ounces	28	grams	g
lb	pounds	0.45	kilograms	kg
	short tons (2000 lb)	0.9	tonnes	t
		VOLUME		
tsp	teaspoons	5	milliliters	ml
tbsp	tablespoons	[illegible]	milliliters	ml
fl oz	fluid ounces	[illegible]	milliliters	ml
c	cups	0.24	liters	l
pt	pints	0.47	liters	l
qt	quarts	0.95	liters	l
gal	gallons	3.8	liters	l
ft^3	cubic feet	0.03	cubic meters	m^3
yd^3	cubic yards	0.76	cubic meters	m^3
		TEMPERATURE (EXACT)		
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

*1 in = 2.54 (exactly). For other exact conversions and more detailed tables, see NBS Misc. Publ. 286, Units of Weights and Measures. Price $2.25. SD Catalog No. C13.10.286.



Approximate Conversions from Metric Measures

Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
mm	millimeters	0.04	inches	in
cm	centimeters	0.4	inches	in
m	meters	3.3	feet	ft
m	meters	1.1	yards	yd
km	kilometers	0.6	miles	mi
		AREA		
cm^2	square centimeters	0.16	square inches	in^2
m^2	square meters	1.2	square yards	yd^2
km^2	square kilometers	0.4	square miles	mi^2
ha	hectares (10,000 m^2)	2.5	acres	
		MASS (WEIGHT)		
g	grams	0.035	ounces	oz
kg	kilograms	2.2	pounds	lb
t	tonnes (1000 kg)	1.1	short tons	
		VOLUME		
ml	milliliters	0.03	fluid ounces	fl oz
l	liters	0.125	cups	c
l	liters	2.1	pints	pt
l	liters	1.06	quarts	qt
l	liters	0.26	gallons	gal
m^3	cubic meters	35	cubic feet	ft^3
m^3	cubic meters	1.3	cubic yards	yd^3
		TEMPERATURE (EXACT)		
°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F



ACKNOWLEDGMENTS

The authors would like to extend their appreciation to Mr. Michael D. Couturier and LT Louis Nash of the U.S. Coast Guard Research and Development Center, and to Dr. Thomas Spence of the Office of Naval Research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (cont'd)

LIST OF FIGURES (cont'd)

LIST OF TABLES

Chapter 1
INTRODUCTION

1.1 BACKGROUND

The Oceanography Branch has, as part of the Improvement of Probability of Detection in Search and Rescue Project, an ongoing investigation into the drift and leeway of survivors and survivor crafts (see Paskausky, 1986). Murphy et al 1982; Anderson, 1984; and Murphy and Allen, 1985; evaluated the Coast Guard's operational search planning computer models. They showed that model predictions of survivors and survivors craft's drift, based on either historical current files or surface current generated by large scale winds from U.S. Navy Fleet Numerical Oceanography Center (FNOC), rarely predicted the actual drift of survivor craft as simulated by freely drifting buoys. At present, a single radio-direction finder type datum marker buoy (RDF-DMB) is used to measure surface currents during a search. The initial position of the buoy is established when the RDF-DMB is dropped from an aircraft. However, to obtain a velocity datum, a second position, at a later time, must be determined for the RDF-DMB. This requires the aircraft to break off the search for the survivors and search for the RDF-DMB, causing the loss of valuable time and fuel. The important velocity measurements necessary for an accurate prediction of the drift of the survivors are therefore limited to one or two points. The development of VHF Loran-C buoys (Allen, Eynon, and Robe, 1987), demonstrate that coded buoys reporting their Loran-C positions every 30 minutes will deliver far more positions, more accurately and therefore more surface current data to the search planner. This provides the search planner a clearer picture of the surface currents. The increased data rates requires a more involved analysis than the data from the RDF-DMB's. This report takes a first look at an analysis technique for freely drifting buoys and other sources of randomly-spaced, surface current data. This technique is

known as objective analysis.

The use of statistical models for the optimal estimation of oceanic fields given limited measurements is known as objective analysis (Bretherton, et al, 1976). This technique has been applied by Gandin (1965) to analyze the wind and pressure field in the atmosphere, and used routinely for the preparation of numerical weather prediction. It is a very important tool in oceanography for both analysis and for observational array design (White and Bernstein, 1979; Clancy, 1983 and Robinson and Leslie, 1985).

The models described here are the extension of space-time objective analysis, and serve as the statistical component of the Harvard Ocean Descriptive Predictive System (Robinson and Leslie, 1985). They can be used as an interpolation scheme both spatially and temporally to provide initial and/or boundary conditions for the dynamic model (Tu, 1981), or used to forecast the oceanic fields (Carter, 1983). The essential assumption in practical use is sufficient statistical knowledge about how the fields are related both in time and space.

In this study, objective analysis was utilized to nowcast/forecast the surface currents using limited drift measurements. Several types of freely drifting buoys that transmit their positions to a shore-based receiver have been used by the Oceanography Branch. A local positioning system using the Microwave Tracking System (MTS) was used with surface drifters in leeway studies. Both polar-orbiting, NOAA/TIROS series satellite and Loran-C positioning were used with surface drogued buoys during drift studies on the continental shelf. All of these studies (Murphy and Allen, 1985; Allen, Eynon, and Robe, 1987; Nash and Willcox, 1988) typically contained data sets in the form of a time series of buoy positions for one to six drifters. A time series of a freely-drifting buoy's positions is called a drift track. The purpose of this study

was to apply the technique of objective analysis to one of our data sets. A computer program using this technique has been successfully applied to large oceanographic data sets of drifter tracks (Robinson and Leslie, 1985). These large data sets, however, included many more drifter tracks than are typically used in our studies. Therefore, the objective analysis program was converted for use on Hewlett Packard microcomputers and applied to a data set from an MTS leeway study.

1.2 BASIC THEORY

The basic equations of the objective analysis procedure are well described in the paper by Bretherton, et al. (1976), to which readers are referred for the details. In this section the special case of scalar objective analysis is described and the general matrix forms, which is applicable to analysis of both scalar and vector fields, are presented.

In scalar objective analysis, the value θ_p of a scalar variable $\theta(x,y)$ at a general point X_p (x,y) is estimated from measurements ϕ_m at data points X_m (x,y), where m = 1, ..., N. A basic assumption of the procedure is that the field θ is stationary and homogeneous and has known mean and covariance. The field statistics are typically estimated from a combination of *a priori* assumptions and a sample of measurements.

Each measured value ϕ_m at point X_m is considered to be the true value θ_m of the field at that point plus some random noise e_m, due to sampling or instrumental error:

$$\phi_m = \theta_m + e_m, \quad (m=1,\ldots,N), \qquad (1)$$

where N is the number of measurements. The errors e_m are assumed to be uncorrelated with the field θ and with one another and have known variance

$$\overline{e_m \phi_n} = 0 \tag{2}$$

$$\overline{e_m e_n} = \varepsilon^2 \delta_{mn}, (m,n = 1,\ldots,N), \tag{3}$$

where the function δ_{mn} has value of unit when m equals n, but zero otherwise. What the above assumes is that there are no systematic or calibration errors.

To reconstruct the entire field the estimator $\hat{\theta}_p$ for $\theta(x,y)$ at each point X_p is found. The Gauss-Markov theorem states that the optimal linear estimator for $\theta(x,y)$ at each position $X_p(x,y)$ is:

$$\hat{\theta}_p = \sum_{m=1}^{N} C_{pm} \left[\sum_{n=1}^{N} A^{-1}_{mn} \phi_n \right], \tag{4}$$

where A^{-1}_{mn} are elements of the inverse of the matrix of covariances between all NxN pairs of observations, and C_{pm} is the covariance between the field value θ_p to be estimated and the nth measurement. The covariances are generally assumed to be a function of the distance between the field positions.

$$A_{mn} = \overline{\phi_m \phi_n} = F(\mathbf{x}_m - \mathbf{x}_n) + \varepsilon^2 \delta_{mn} \tag{5}$$

$$C_{pm} = \overline{\theta_p \phi_m} = F(\mathbf{x}_p - \mathbf{x}_m) \tag{6}$$

For given estimation and observation positions, A_{mn} and C_{pm} are constants. Equation (4) therefore expressed the estimate for the field variable as a linear combination of the measurements, weighted by the covariances.

The minimum mean square error for the estimator is the variance of the error in $\hat{\theta}_p$ given by:

$$\overline{(\theta_p - \hat{\theta}_p)^2} = C_{pp} - \sum_{m,n=1}^{N} C_{pm} A_{mn}^{-1} C_{pn}. \qquad (7)$$

The first term is the overall natural variation in the field θ, while the second term is a measure of information provided by the measurements.

A more general vector form for equations (4) and (7) is:

$$\hat{\theta} = C_{\phi\theta} C_{\phi}^{-1} \phi \qquad (8)$$

$$C_e = C_{\theta} - C_{\phi\theta} C_{\phi}^{-1} C_{\phi\theta}^{T}, \qquad (9)$$

where $\hat{\theta}$ is the estimated vector describing the entire field. $C_{\phi\theta}$ is the cross-covariance matrix between the field vector and the observation vector. C_{ϕ} is the auto-covariance matrix for the vector of observations ϕ. C_{θ} is the natural covariation of the field as defined by the a priori covariance function. C_e is the error variance. The superscript T denotes the matrix transpose.

Equations (8) and (9) are general and can be applied to a two-dimensional or higher vector variable. For example, given two component (u,v) observation of velocity at N points, the covariance matrix C_{ϕ} now includes the auto-covariances of each observational component and cross-covariances between the two components. The matrix $C_{\phi\theta}$ includes the covariance between each observation component and each component of the estimated field variable. Vectors $\hat{\theta}$ and ϕ include values for each component at each estimation and observation point, respectively. It should be pointed out that the theorem does allow replacing the covariance by the correlation functions in practical use.

Chapter 2
METHODS

2.1 INTRODUCTION

The covarinace matrices required in the solution of the objective analysis as described in the Basic Theory section are defined by correlation functions. The correlation function is the planar (x,y) and temporal (t) distribution of the weighting factors for estimating an unknown value from the surrounding observations. Essentially, closer observations are weighted more in determining an unknown value than more distance values. At sufficiently large distances and time lags the observations will have no appreciable influence on the interpolated value. Thus, a limit over which the observations are not included in the interpolation is assumed. This greatly simplifies the computation. The correlation function can be determined directly from a data set that is sufficiently large. However, when the data set is small, the mathematical form of the correlation function is artificially imposed.

2.2 THE DETERMINATION OF CORRELATION FUNCTION

The essential assumption in objective analysis is that the correlation functions are known. These functions can be calculated from the measurements by statistical approaches, if there are large and intensive data sets available. The determination of these correlation functions are very difficult when the observed data are sparse. An alternate approach is to use a fitted analytical formula which represents the correlation function of the limited data. The correlation function should satisfy the following two requirements: (1) symmetric to the lag, r, i.e., C(r)=C(-r), where r = (delta_x,delta_y,delta_t) is the space-time lag, and (2) in positive definite form. It is important to note that the calculated correlation function may contain some small scale noise which must be filtered. This

smoothed correlation function is then used as a "look-up table" for the objective analysis.

2.3 THE ELIMINATION OF DISTANT DATA

The correlation function drops off with increasing space and time lags. In other words, the distant (in space and time) observations have very little influence on an interpolation point when compared to nearby points. So the elimination of the distant observations could make the computation very efficient. Usually, limiting the domain of influence (in space and time) of an interpolation point is sufficient to reduce the matrices used in the analysis to a reasonable size, but with a large data set the limited influence domain may still contain a lot of data. For the large data case, the maximum influence points are "limited" to less than the full number of data points in the domain. The data points which have the highest correlation with the interpolation points or the points that individually give the lowest error estimate are ordered by rank, and then the top "limited" values are used in the calculation. These are the optimal data points.

2.4 THE CORRELATION FUNCTION USED IN THIS STUDY

The correlation function used in this present study is:

$$C(R) = e^{-R}$$
$$\text{where } R = \sqrt{\left[\left(\frac{\Delta x}{x\,\text{fold}}\right)^2 + \left(\frac{\Delta y}{y\,\text{fold}}\right)^2 + \left(\frac{\Delta t}{t\,\text{fold}}\right)^2\right]} \qquad (10)$$

where delta_x, delta_y, and delta_t are the components of the difference from the data point to the interpolated point; and where xfold, yfold, and tfold are the e-folding scales of the correlation function.

Equation (10) for x and y is shown in Figure 1, for values of xfold, and yfold equal 1. At x,y = 1,0 and x,y = 0,1 the

correlation (z) value is 0.368, this is called the e-folding scale. The e-folding scale is shown in Figure 1 as the circle at z = 0.368. As the correlation function approaches zero asymptotically, the outer portion is cut off by the limit set in Equation (6). The appropriate values for the e-folding scales are the physical scales of the motions. This is the scale in time and space of the coherence of the dominant motions. The e-folding scale is symmetrical about its axis, but does not have to be equal about all axes. Figure 2 is the correlation function where xfold = 1 and yfold = 3. In Figure 3 the xfold = 3 and yfold = 1, in these cases the e-folding scale are ellipses at z = 0.368. These simple examples can be transferred to a more realistic case, e.g., the field in question is alongshore velocity on the continental shelf, the coherence is very long (~200km) in the along shore (y) direction and short in the cross shelf (x) direction (~20kms) and about two days in (t) time. These values could be used as a first guess of the e-folding scales for x, y and t. The appropriate values are based on the physical scales of the value used.

2.5 PROGRAM DESCRIPTIONS

The SOA/VOA (Scalar/Vector Objective Analysis) Programs computer software package consists of twelve programs and related documentation, which were originally written by Carter (1983) in FORTRAN language for the VAX series mainframe computers. They were rewritten by Dr. Sun, in Hewlett Packard (HP) BASIC and designed to run on Hewlett Packard 9000 series 200 mini-computers. Many enhancements were added to make the programs easier to use (for example, better prompts for inputting the parameters and outputting the results, easier error checking on inputs, etc.). The distribution disc labeled "DISC_1" contains the twelve programs and related documentation.

The package contains programs for displaying the original data, setting up the objective analysis, both a scalar and a vector

analysis program, and programs for displaying and analyzing the results of the objective analysis. There are three programs that display the data from the drifters. The u and v components of velocity can be plotted as a time series, or individually on the x,y plane. The PREPARE and GET_DATA programs set up the drifter data for input into the main objective analysis programs. The GET_IP programs converts the original coordinate system to the interpolated positions of the array used in the objective analysis. Either of two programs provide the correlation function for the objective analysis. COM_COR calculates the correlation function from the observed data set and the subroutine FNCOV_F calculates the function from a given mathematical formula. Either the Scalar Objective Analysis (SOA) or the Vector Objective Analysis (VOA) are the main programs for determining the interpolated fields. Working programs are stored under the names SOA_W and VOA_W. Two programs are for displaying the results of the objective analysis. PRINT_OA shows the individual numeric values on the CRT and prints them on the printer. PLOT_OA draws the contours of the results and displays them on the CRT or prints then on a printer or plotter. The difference between two fields generated by separate runs of the objective analysis is calculated by the program SOA_DIFF.

A brief description of the programs follow.

The three programs that display the data from the drifters are:

LINDAT_UV PROGRAM:
plots the u- and v-component of the current speeds as a function of time. The u- and v-speed are indicated as solid and dotted lines respectively.

PLOT_PUS PROGRAM:
plots the drifter position and the u-component of the current speed at a specific time.

PLOT_PVS PROGRAM:

plots the drifter position and the v-component of the current speed at a specific time.

The three programs that set up the drifter data and the interpolated array for input into the main objective analysis programs are:

PREPARE PROGRAM:

prepares the observation data for test run by converting the original data (x,y,t) to data files by drifter of x,y,t,u, and v.

GET_DATA PROGRAM:

reads in the observed data, and sets up the input and output files for the objective analysis program.

GET_IP PROGRAM:

converts the original coordinate system (e.g. Florida state plane) to inter/extrapolation positions of the objective analysis array.

Two programs provide the correlation function for the objective analysis, COM_COR program and FNCov_t subroutine or by the FNCov_f subroutine.

COM_COR PROGRAM:
calculates the correlation function from the observed data, and stores them in the form of a "look-up" table.

FNCov_t SUBROUTINE:

gets the correlation function from the "look-up" table.

FNCov_f SUBROUTINE:

calculates the correlation function from a fitted analytical formula.

SOA/VOA PROGRAMS:

The "SOA" and "VOA" programs play the key roles in the scalar and vector objective analyses respectively. Each program is composed of one main program and several sub-programs. The main program prompts for inputting parameters from keyboard and reads in the measurements and inter/extrapolation position data, calls sub-programs to do the objective analysis, then finally outputs the estimated fields and expected errors to disk in Drive No. 1.

SOA_W PROGRAM:

The working program of the scalar objective analysis.

VOA_W PROGRAM:

The working program of the vector objective analysis.

Two programs display either the individual numerical results or the contours of results.

PRINT_OA PROGRAM:

displays the individual results of the objective analysis results on the screen. It also outputs the results to the printer, if desired.

PLOT_OA PROGRAM:

contours the objective analysis results and then plots to the screen. It also dumps the graphics to the printer or plotter, if desired.

SOA_DIFF PROGRAM:

calculates the difference between two 15 by 15 arrays (File 1 - File 2) and determines the Cumulative Measured Error Index (Equation 9).

A brief description of the major subprograms of the SOA/VOA program follow.

DIAG SUBPROGRAM:

calculates the statistical parameters of inputs such as the mean, variance, root mean square, minimum and maximum.

EST_MEAN SUBPROGRAM:

calculates the estimated mean, and then removes the estimated mean from the input array.

FNIER SUBPROGRAM:

returns error code for subprogram INVMTX.

GET_RD SUBPROGRAM:

gets the data before and after a given time and also within a given spatial radius from the domain reference point.

INVMTX SUBPROGRAM:
inverts the Matrix.

SCALAR_OA SUBPROGRAM:

The scalar space-time objective analysis routine.

SELECT SUBPROGRAM:

eliminates the distant (in space and time) data points and selects the nearest data points to an inter/extrapolation point X,Y, and T. The number of optimal data points is restricted to the maximum number of influential points.

SET_INVA SUBPROGRAM:

sets up the correlation function for the input data given the optimal positions and times, it returns the inverted correlation function matrix.

SORT SUBPROGRAM:

sorts the index and correlation in descending order.

VECTOR_OA SUBPROGRAM:

The vector space-time objective analysis routine.

2.6 USER'S INSTRUCTIONS AND EXAMPLES

In this section the reader is shown how to run the programs by using the realistic field data.

The procedures to run the SOA programs are as follows: (note: hit <ENTER> key, after you answer the program prompts.)

Step 1: Boot the system.

Step 2: Insert the distribution disk labeled DISC_1 into Drive No. 0.

Step 3: Insert the data disk labeled DISC_2 into Drive No.1.

Step 4: Prepare the observational data.

The drifter position data named "LDI01APR" in DISC_2 is used here as an example. The "LDI01APR" contains positions of ten drifters and four boats as a function of time. The program PREPARE calculates the drifting u- and v-speed for each drifter and boat, and stores the drifter positions, times and calculated current components by using the format of "X,Y,T,U,V". Files Drifter_1 to Drifter_10 are the drifter data, while Files Drifter_11, 12, 13, and 14 are the drifting speed data for boats Little Lady, Whaler, Joanna, and R/V Oceaneer IV (drifter) respectively.

The inter/extrapolation of the u-speed from the drifter data is the objective. The program GET_IP calculates the area of interest which is a square domain with 15x15 grid points. The inter/extrapolation position data are saved in file IP_POS. The correlation function used in our present study is equation (10).

The first test run used data from 6 drifters including Drifters No. 5, 6, 7, 8, 9, and 10. The GET_DATA program was used to get these drifter data and store them in file "OBSDATA6". Different runs can be made by changing the file names in the "CREATE BDAT" lines 1042, 1043, 1044, and the "ASSIGN @Pathout" line 1047 and commenting out the appropriate drifter "DATA" lines (1015-1040) and changing the upper limit of "N" in line 1050 (FOR N = 1 to n).

Step 5: Get a working program.

1. Insert the distribution disk labeled DISC_1 into Drive no. 1
2. Load "SOA" program. Type LOAD "SOA", hit <ENTER> key.

3. Load "FNCov_f" or "FNCov_t" subprograms. Type LOADSUB "FNCov_f"(or FNCov_t"), hit <ENTER> key.

4. Make correction on line 4655 (ENTER @Path_in; X, Y, T, Phi, Dummy) for observed data format, and line 4775 (ENTER @Path_in; X, Y) for inter/extrapolation position data format, if needed.

<u>Step 6:</u> Store this working program for future use, if desired. Type STORE "SOA_W", hit <ENTER> key.

<u>Step 7:</u> Now, you are in a position to run the Scalar Objective Analysis and interested in inter/extrapolation the field on 14:30:00, 1 April 1985. Hit <ENTER>/<RUN> key to start, the program will prompt

"Do you need the documentation? (answer Y/N, for yes/no)"

Type "N" for no documentation, and begin the computation.

1. "Enter time the analysis to make. (DD MMM YY,HH:MM:SS)"
   Type 1 APR 1985,14:30:00

2. "Enter number of objective analyses to make."
   Type 1

3. "Enter time interval for extrapolation in time."
   Type 0

4. "Enter the maximum distance radius from the reference point of domain."
   Type 1E+6

5. "Enter the maximum time radius before and after the time of the analysis."
   Type 0

6. "Enter the maximum space and time lags."
   Type 5E+4,0

7. "Enter the maximum number of influential points."
   Type 6

8. "Enter the X direction e-folding Scale."
   Type 1E5

9. "Enter the Y direction e-folding Scale."
   Type 1E5

10. "Enter the Time e-folding Scale."
    Type 1000

11. "Enter the observed data file including file specifier."
    Type OBSDATA6:,700,1

12. "Enter the interpolation position file including file specifier."
    Type IP_POS:,700,1

13. "Enter the file specifier for OA forecast fields."
    Type SOA_FCST6U:700,1

14. "Enter the file specifier for OA error variance fields."
    Type SOA_EVAR6U:,700,1

Step 8: The working program "SOA_W" will output the forecast field and expected errors fields to Disc_2 in Drive #1.

Step 9: Output the results. Use programs PLOT_OA and PRINT_OA to get the hard copy of the analysis results.

Repeat steps 5-9 to calculate the v-component speed, after making changes to line 4655 of the SOA_W program (switch the positions of the "Phi" and "Dummy" variables).

Additional comments on the responses to the 14 questions of Step 7 follow:

1. This is the start time.
2. More than one objective analysis will produce a series of forecasts.
3. This is the interval in seconds from the start time (1.) to the first forecast and each forecast thereafter. The example used is for a single nowcast. For a forecast, the analysis will occur first at time (1. +.3.) and continue for (2.)
4. and 5. The limits in 4. and 5. are for the full data set that will be used in the objective analysis. The units are meters for 4. and seconds for 5. To include data both before and after the start time increase 5. In this case, data is sampled every 2 minutes or 120 seconds, therefore by setting 5. to 120 the available data goes from 6 values at 14:30:00 to 18 values at 14:28:00, 14:30:00, and 14:32:00.
6. This is limit in meters and seconds for the elimination of the distant observations from each calculation. These values should be smaller than 4. and 5.
7. The maximum number of influential points used for each calculation and within the limits of 6. As this values becomes large more outlining data point are used. This greatly increases the computer time required, without a significant increase in the quality the final result. With too few influential points (certainly 3 or less) the nearest values are only used and information from other close points are lost to the calculation. At the limit of 1, the single closest point is used, resulting in a field that is a series of "plateaus".

8., 9., & 10. The correlation function used here falls off from 1 at zero by e to -R power, where

$$R = \text{sqrt}[(\text{delta\_x}/\text{xfold})^2 + (\text{delta\_y}/\text{yfold})^2 + (\text{delta\_t}/\text{tfold})^2].$$

11. and 12. The two input files. The data file and the file of the interpolated positions.
13. and 14. The two output files. This first is the forecasted field of the values and the second is the error field. The expected error field is a measure of the confidence at each interpolated point. The closer the data point are to the interpolated point the greater the confidence and smaller the expected error.

Chapter 3
TEST RUNS

3.1 INTRODUCTION

Objective analysis has been successfully applied to large data sets of freely drifting oceanographic buoys. However, what occurs to the results when the data available to the objective analysis program is systematically reduced to the limit of a single available point? Does the technique fail catastrophically, or do the results simply degrade with fewer data available for input? And if the results just degrade, how do they degrade? To answer these questions, a series of test runs of the scalar objective analysis (SOA) program were done. The data set used to test the SOA was from the Fort Pierce, FL, March-April 1985 Leeway Experiment. During the afternoon of 1 April, six surface drifters were deployed and were tracked with the Microwave Tracking System (MTS). Lieutenant Louis Nash, USCG, provided the edited u and v velocity data in the x-y state coordinate plane, Figures 4-9. Thus we had a data set which was large by our standards for investigating the SOA program. To further simplify the problem we only used velocities at a single point in time - 14:30.

3.2 TEST RUNS

During the test runs of the objective analysis program we first tried to calculate the correlation function using the COM_COR program on all six drifters. However, six drifters proved to be insufficient for the calculation of a sensible correlation function. Therefore we used the analytic form of an exponential decay function as given by equation (10), the correlation function.

Equation (10) was used as the basis for generating current field values on a grid of 15 by 15 regular spaced points from the observed surface current data. That grid has points between current data points and to a limited extent outside the region. The question was then asked, "What is the minimum number of drifters needed by the SOA program to produce a sensible current field?". To start, all six drifters were used to define a "best" estimate of the existing current field. This provided the best estimate of the u-current field against which the other fields were compared. When drifters were removed one at a time from the data set, the forecasted fields changed. The fields generated by the reduced number of drifters were then compared to the results with the "best estimate" field determined by all six drifters. Two different schemes for removing drifters were investigated. Case "A" was to remove the **least** useful drifters one at a time leaving the "best" to be used to estimate the U-field. Case "B" was to remove the **most** useful drifters one at a time leaving the "worst" to provide the estimate of the U-field. Both schemes are shown in Table I.

TABLE I - DRIFTER CASES

CASE-A (BEST)	DRIFTERS (USED)	CASE-B (WORST)	DRIFTERS (USED)
6	5, 6, 7, 8, 9, 10	6	5, 6, 7, 8, 9, 10
5A	5, 7, 8, 9, 10	5B	5, 6, 7, 9, 10
4A	7, 8, 9, 10	4B	5, 6, 7, 9
3A	7, 8, 10	3B	6, 8, 9
2A	8, 10	2B	6, 9
1A	8		

TABLE I: The Best (A) and the Worst (B) cases used for comparisons of subsets of drifters with the full set of six drifters.

The differences were calculated at each 15 by 15 grid point between the forecasted field derived from all six drifters and each 15 by 15 grid point of the cases above. The differences from all the 225 points produced the measured error fields. The square root of the sum of the squared differences is an index of the cumulative measured error. The equation for cumulative measured error index for case 5A is shown below:

$$\text{Cumulative Measured Error Index} = \sqrt{\Sigma}\,(6_{ij} - 5A_{ij})^2 \qquad (11)$$

where the subscripts "ij" are the elements of the 15 by 15 array. Therefore, we have fields of the forecasted U-velocity, expected error, measured error and an index of the cumulative error.

Figure 10 shows the eastward component of velocity (u) at 14:30 from the six drifters and Figure 11 is the northward (v) component of velocity. In Figure 12 the current field generated by the SOA scheme for the six drifters is shown. The currents

near the drifters are reproduced faithfully enough, including the region of currents less than 10 cm/s in the upper central region and the currents greater than 16 cm/s in the southwest corner. Figure 13 shows the expected errors and they are low across the region with slightly higher values in the four corners, as expected. The measured error for the 6 drifters case is zero by definition, and therefore is not shown. The objective analysis forecast of the U-fields for scheme "A" are presented in Figures 14 - 17. The expected error generated by the objective analysis program for the "A" scheme are shown in Figures 18 - 21. The fields of measured error for scheme "A" are shown in Figures 22 - 25. The "B" scheme forecast fields, expected error fields and measured error fields are presented in Figures 26 - 37.

The forecast fields for all scheme "A", Figures 14 - 17, maintains the basic field forecasted by the full six drifters, Figure 12. This is also shown by the measured error fields, Figures 22 - 25, which are generally low right across the field. Even "2A" which uses just drifters #8 and #10 reproduces the U-field without trouble, Figures 17 and 25. This is in direct contrast with scheme "B" forecast fields, Figures 26 - 29. Here we can see that even with five drifters the forecasted field differs greatly from the original field, Figure 12. The measured error fields confirm this, Figures 34 - 37.

The cumulative measured error index for the two schemes are presented in Table II.

TABLE II.

CUMULATIVE MEASURED ERROR INDEX

Number of drifters used for Objective Analysis						
	6	5	4	3	2	1
Scheme A	0.0	3.2	5.7	13.3	14.9	49.4
Scheme B	0.0	14.7	18.9	27.3	24.2	--

TABLE II. The Cumulative Measured Error Index for Scheme A and Scheme B.

This suggests that "2A" did as good a job as predicting the U-field as did "5B".

Scheme "1A" with just drifter #8 is the special case of the objective analysis taken to its limit. The forecast field, Figure 38, is uniform at the value of the drifter (9.028 cm/s). The expected error field, Figure 39, increases away from the location of the drifter due to the decrease in the correlation function, Equation 11. The measured error field, Figure 40 has the same form as the original forecast field for the 6 drifter (Figure 12), but offset by 9 cm/s.

3.3 RESULTS OF TEST RUNS

The main portion of the objective analysis technique did not fail catastrophically. Instead, the resulting velocity fields decreased relative to the original field, reading from left to right - Table II. However, the more important point is that the placement of the buoys or drifters relative to the flow field features will greatly influence how well the total flow field is estimated, reading from top to bottom - Table II. An example of an application would be the placing of two buoys on either side of a oceanic front, estimating the currents better than several

buoys all on the same side of the front. A few well placed buoys will provide better information than many poorly placed buoys.

The determination of the correlation function (Section 2.2) from the statistics did fail for the data set used here. The number of buoys required to determine the correlations functions is greater than six. Therefore an analytical formula (Equation 10) was used instead. In this study neither the v-velocity (north - south) field nor the time dependent field were considered. Since there should be a relationship between the u and v fields and between the spacial and temporal fields, correlations in time could be used to infer correlations in space. Thus, the buoys' track histories could be used to determine the time correlation which then would provide a better estimate of the velocity field. Further work is being conducted on improving estimates of the current field from limited amounts of drifter data.

CHAPTER 4
CONCLUSIONS

4.1 CONCLUSIONS

The search planner requires real-time surface current information to determine the most probable location of the survivors and survivor craft. The displacement with time of the search datum can be estimated from the track histories of freely-drifting surface buoys. All types of freely drifting buoys will generate data sets that consist of a time series of irregular-spaced positions. As a set of buoys freely drift in the ocean, (1) positions at any one time are not on a regular grid, (2) the displacement from the previous positions of that buoy to the next position is uneven, and (3) the relative displacement among all the buoys is changing with time. Additionally, some types of buoys (e.g. TIROS satellite track buoys) positions are reported intermittently. The results is a "messy" data set.

The results of the test runs suggest that a few well placed buoys can provide a better estimation of the current field than many poorly placed buoys. Therefore, the use of remote-sensing information (e.g., Side Looking Airborne Radar (SLAR) derived sea surface roughness or NOAA6 satellite Advanced Very High Resolution Radiometer (AVHRR) derived sea surface temperature imagery) will aid in the determination of the optimal placement of buoys.

To take advantage of computer analysis of the buoys tracks, the data set must first be cleaned up. The essence of objective analysis is that, given an irregular-spaced data set in time and space, a data set is interpolated and extrapolated to a regular (even and square) grid. From this many standard computer programs are available that can be brought to bear on the analysis of the buoy tracks. This analysis can then provide the most information in a timely manner for the search planner.

In Figure 41 the Search and Rescue Problem is blocked out (A) and the Datum Movement components of SAR Planning are expanded in (B). As the available Environmental Data sets grow in number, size and complexity, the Coast Guard will have to provide a system to handle these data sets. Figure 41(B) illustrates the role Objective Analysis would play in a possible future system. Environmental Data in the form of winds (e.g., FNOC, remotely reporting winds from ships, meteorological-buoys, weather stations, and Next Generation RADAR) and surface currents (e.g., Loran-C and ARGOS buoys, and satellite imagery) would have to be collected and processed initially before passing to the Objective Analysis module. Then the Objective Analysis module would assimilate and convert these irregular spaced data sets into standard arrays. The standard arrays of wind would provide nowcasts of the Wind Field for the Leeway Calculation. The standard arrays of the surface currents could either be the nowcast of the Ocean Current Fields or the Initial

conditions required by the numerical model of the local physical oceanography. The numerical model would then provide forecasts for the Ocean Current Field. The Ocean Current Field along with the Leeway Calculation would be used for Datum Movement calculations.

In summation, an objective analysis program was modified and tested to demonstrate its capabilities on small data sets. The results of the test show that objective analysis can effectively work on as little as two buoy drift tracks. The Coast Guard is going to utilize a variety of environmental data sources to effectively locate the most probable positions of survivors. The objective analysis module will be a first step in preparing these data sets for use by the search planner. The application of these data sets to other computer programs will provide the search planner with the most probable positions of the survivors. Work is continuing on the objective analysis module.

4.2 FUTURE WORK

Presently the entire front end and back end of this set of programs contain room for improvements. Work has started to greatly increase the ease of inputting diverse data sets, including data sets where positions are given in latitude and longitude. Further work is planned to use objective analysis for a variety of data sets: (1) the wind velocity fields using input data from several surrounding weather stations, (2) surface current fields from NOS tidal data, (3) results of local numerical models, (4) satellite AVHRR sea-surface temperature imagery data, (5) sea surface currents determined from successive AVHRR imageries, and (6) the relatively large buoy data set collected by the Commander, International Ice Patrol (CIIP). The CIIP data set will provide experience with a data set similar to what a Group or District would collect after several years of deploying VHF Loran-C or TIROS type datum marker buoys.

REFERENCES

Anderson, I., 1984. An Evaluation of Computer Assisted Search Planning Using TIROS Oceanographic Drifter Tracks. Unpublished Manuscript. U.S. Cost Guard International Ice Patrol, Avery Point Groton, Connecticut 06340, 11pp.

Bretherton, F.P., R.E. Davis, and C.B. Fandry, 1976; A Technique for Objective Analysis and Design of Oceanographic Experiments Applied to MODE-73, Deep Sea Res., V.23, pp. 559-582.

Carter, E.F., 1983; The Statistics and Dynamics of Ocean Eddies, Ph.D. Thesis, Division of Applied Sciences, Harvard University.

Clancy, R.M., 1983; The Effect of Error Correlations on Objective Analysis of Ocean Thermal Structure, Deep Sea Res., V.30, No.9A, pp. 985-1002.

Gandin, L.S., 1965; Objective Analysis of Meteorological Fields, Israel Program for Scientific Translations, Jerusalem, 242 pages.

Murphy, D.L., and A.A. Allen, 1985. An Evaluations of CASP Drift Predictions near the New England Shelf/Slope Front. Report CG-D-16-85. U.S. Coast Guard Office of Research and Development, Washington, D.C.,209593, 51pp.

Murphy, D.L., L. Nash, D.F. Cundy, and S.R. Osmer, 1982. An Evaluation of SARP Drift Prediction Using Satellite-Tracked Drift Buoys. Report CG-D-05-83. U.S. Coast Guard Office of Research and Development, Washington, D.C., 20593, 67pp.

Nash, L., and J. Willcox 1988. Spring 1985 Leeway Experiment. (In Preparation). U.S. Coast Guard Research and Development Center, Avery Point, Groton, Connecticut 06340.

Paskausky, D.F. 1986. Surface current Real-time Prediction for Search and Rescue, Proc. Offshore Technology Conf. 1986, Houston, Texas, pp. 499-501.

Robinson, A.R. and W.G. Leslie (1985). Estimation and prediction of oceanic eddy fields, Prog. Oceanogr. 14, 485-510.

Tu, K., 1981; A Combined Statistical and Dynamical Approach to Regional Forecast Modeling of Open Ocean Currents; Ph.D. Thesis, Division of Applied Sciences, Harvard University.

White, W.B. and R.L. Bernstein, 1979; Design of an Oceanographic Network in the Midlatitude North Pacific, J. Phys. Ocean. V.9, pp. 592-606.

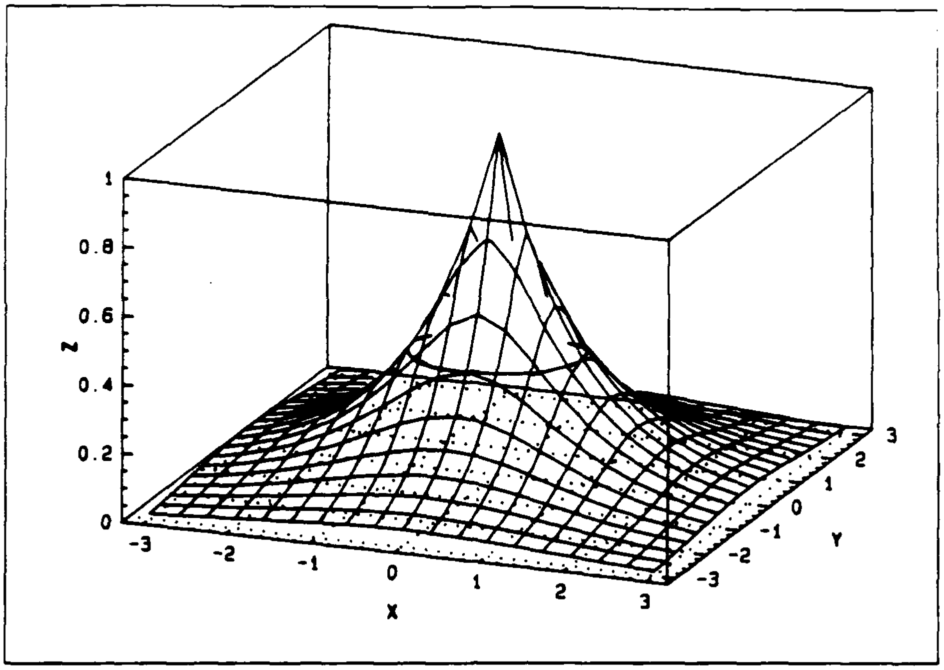


Figure 1. The defined correlation function (equation 10), where z is the value of the function and xfold and yfold =1. The circle at z = 0.368 is the e-folding scale.

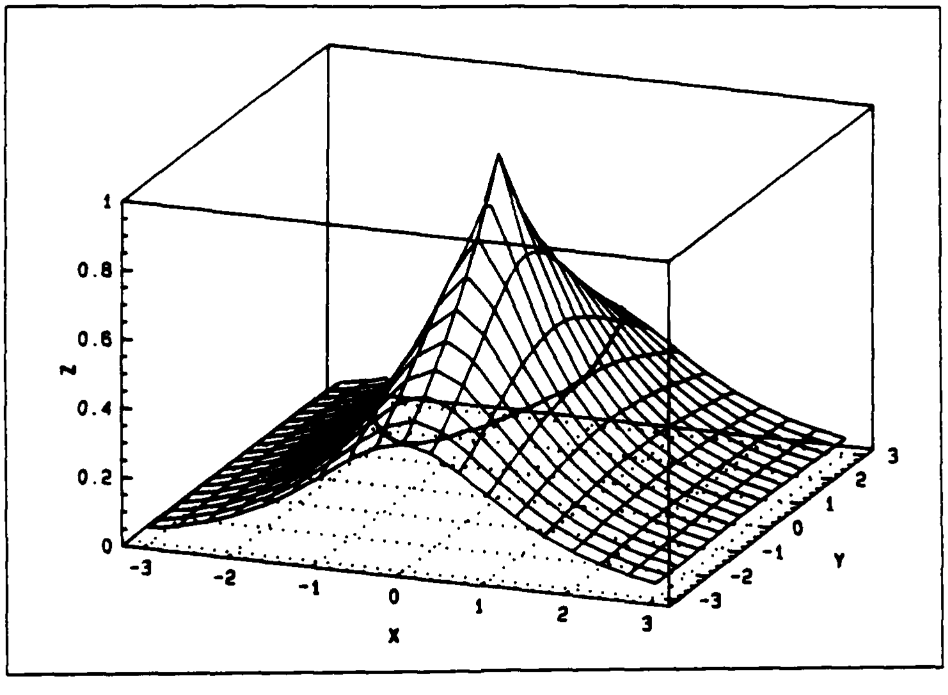


Figure 2. The defined correlation function (equation 10) where z is the value of the correlation function and xfold = 1 and yfold = 3. The ellipse at z = 0.368 is the e-folding scale.

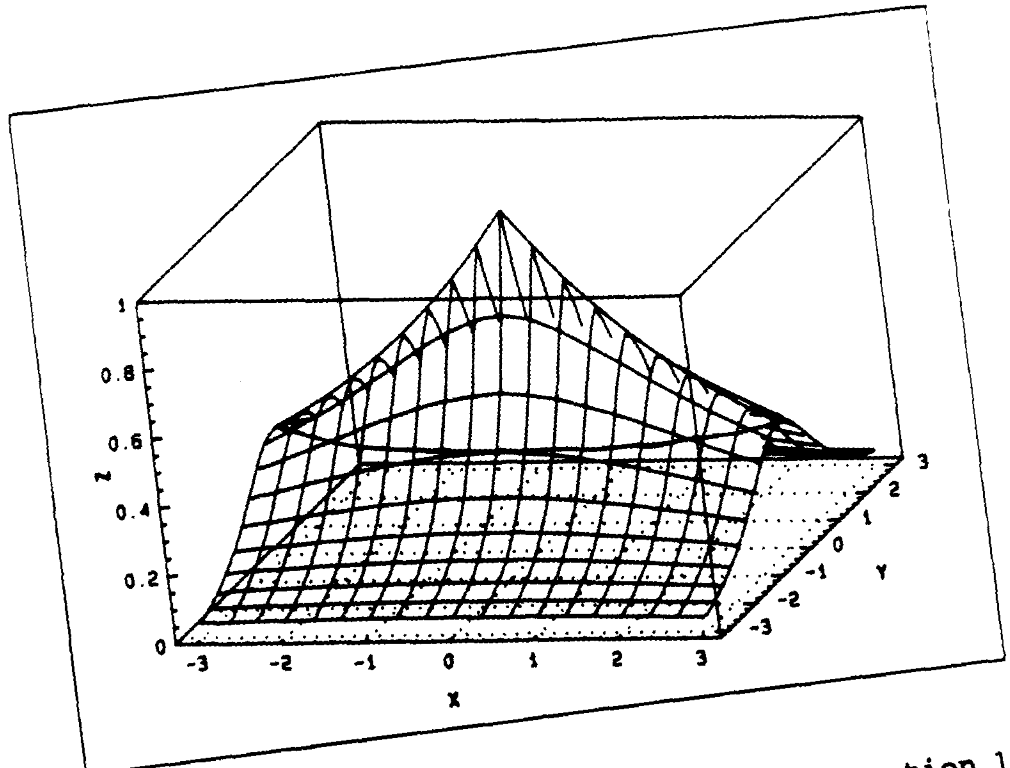


Figure 3. The defined correlation function (equation 10) where z = the value of the correlation function and xfold = 3 and yfold = 1. The ellipse at z = 0.368 is the e-folding scale.

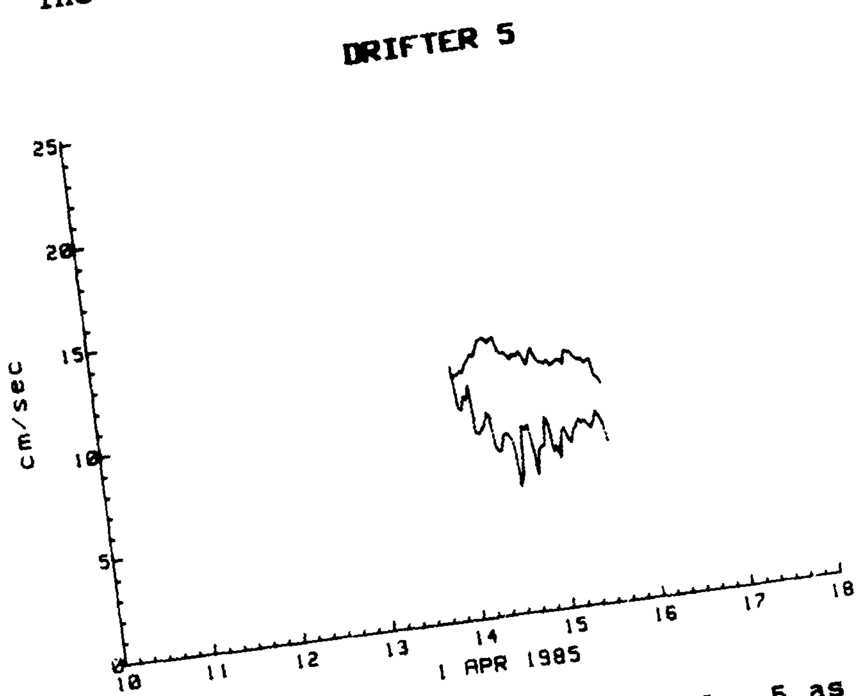


Figure 4. The current speeds of Drifter No. 5 as a function of time. The solid and dotted lines indicate the u- and v-components of current speeds respectively.

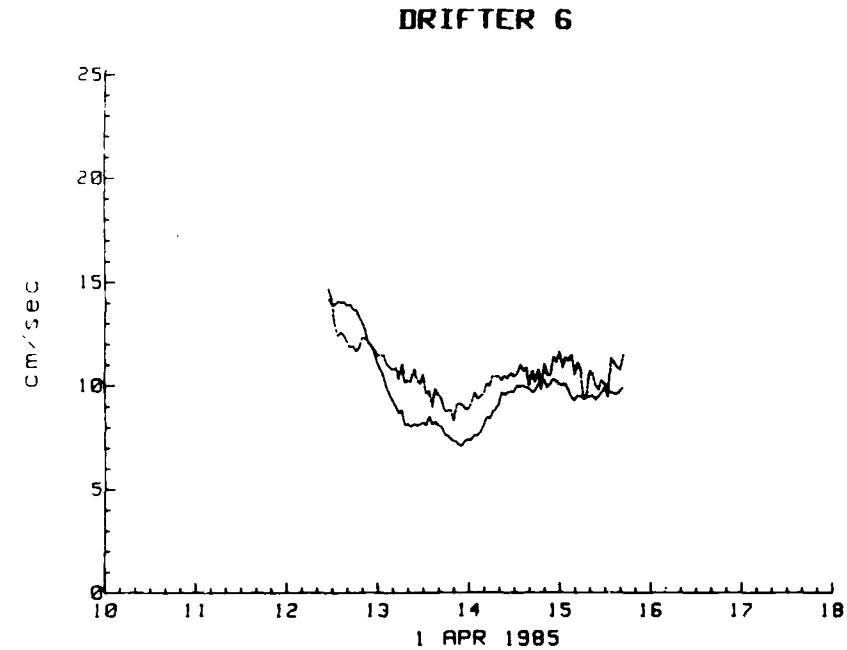


Figure 5. The current speeds of Drifter No. 6 as a function of time. The solid and dotted lines indicate the u- and v-components of current speeds respectively.

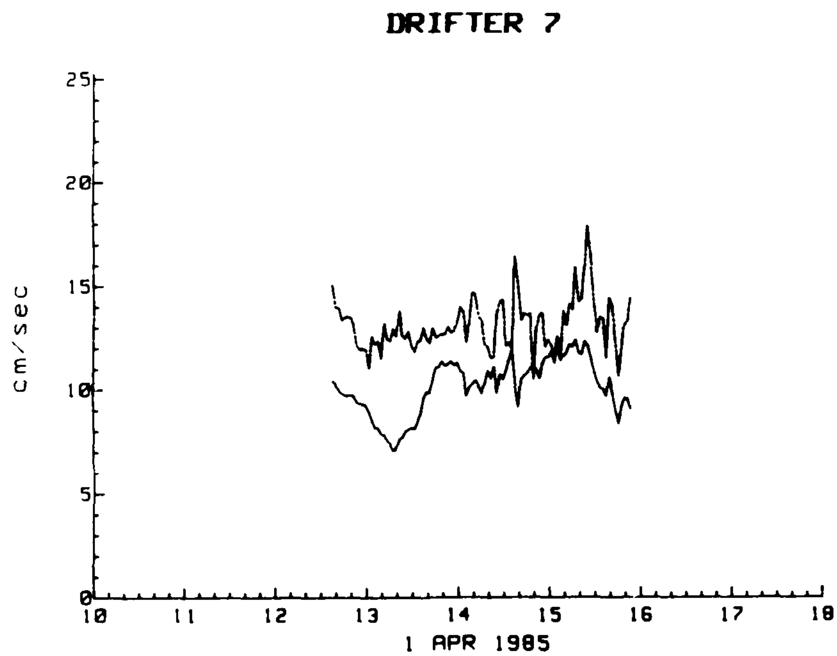


Figure 6. The current speeds of Drifter No. 7 as a function of time. The solid and dotted lines indicate the u- and v-components of current speeds respectively.

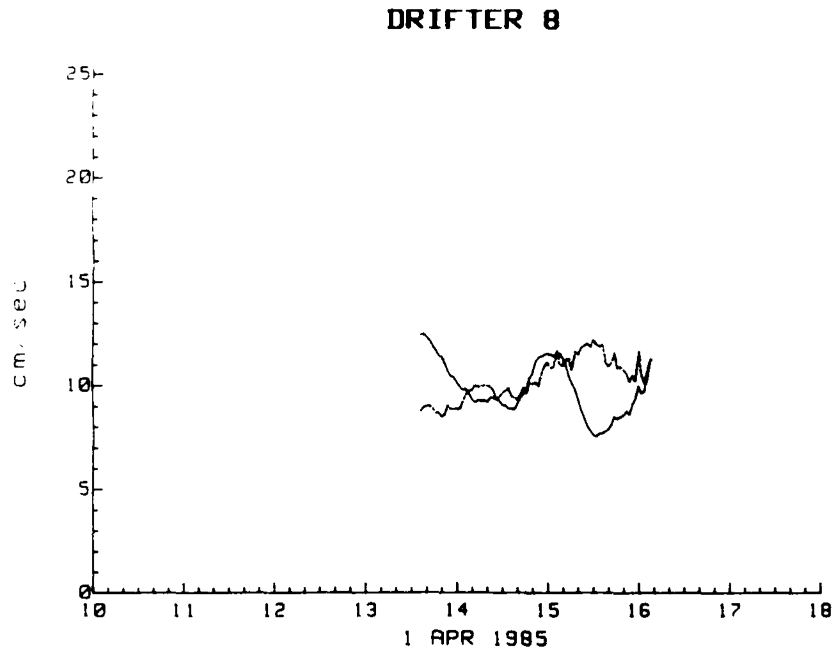


Figure 7. The current speeds of Drifter No. 8 as a function of time. The solid and dotted lines indicate the u- and v-components of current speeds respectively.

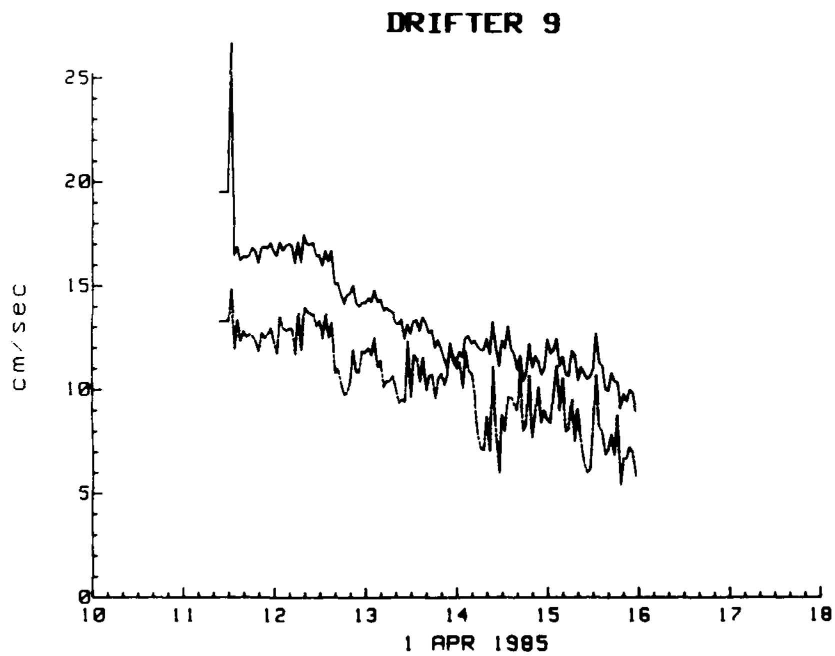


Figure 8. The current speeds of Drifter No. 9 as a function of time. The solid and dotted lines indicate the u- and v-components of current speeds respectively.

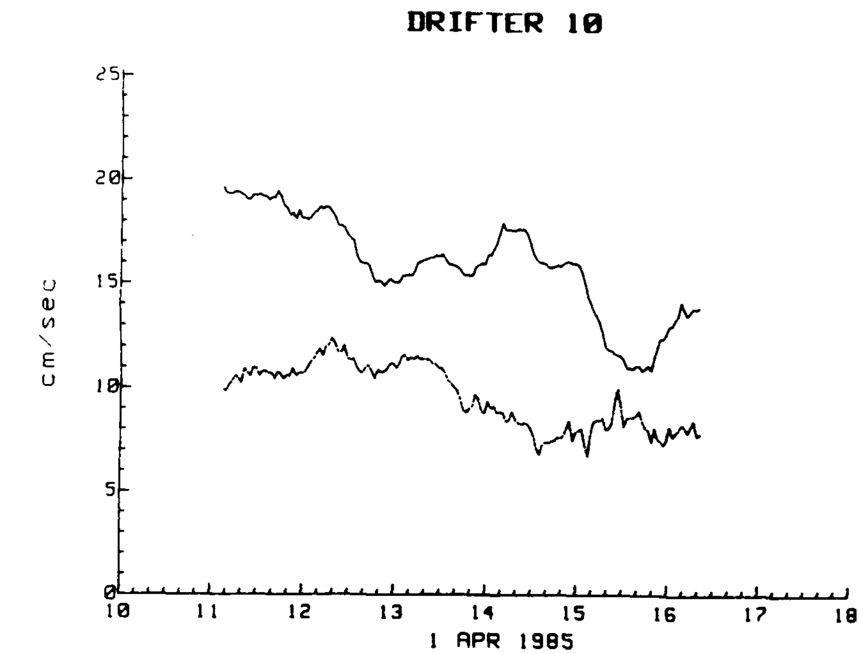


Figure 9. The current speeds of Drifter No. 10 as a function of time. The solid and dotted lines indicate the u- and v-components of current speeds respectively.

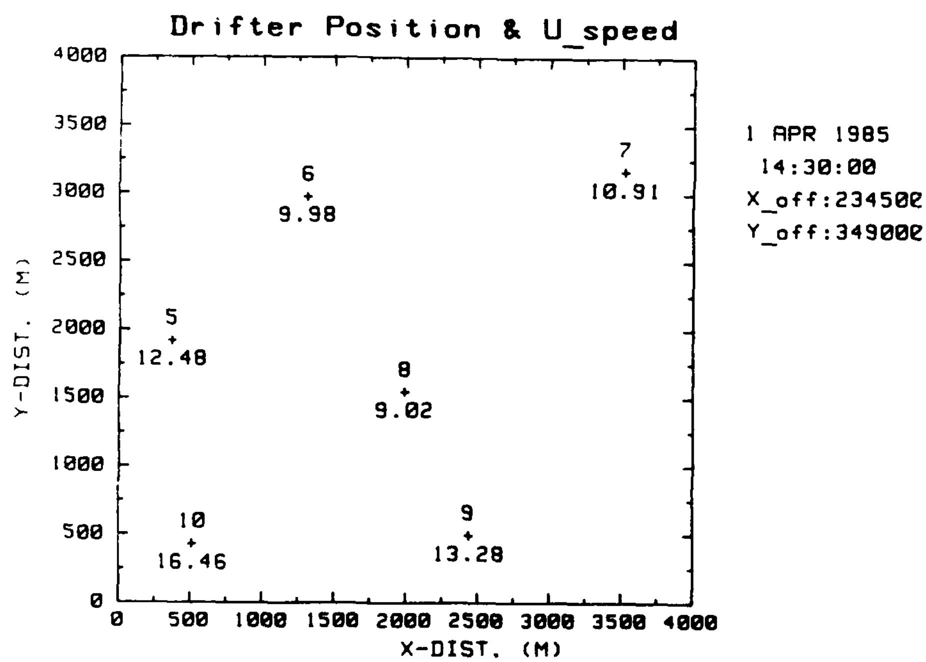


Figure 10. Position and u-component speeds at 14:30, 1 April 1985. The symbols "+" indicate the drifter location. The numerics above the "+" denote the drifter ID; while the numerics below "+" denote the values of the u-component (cm/sec).

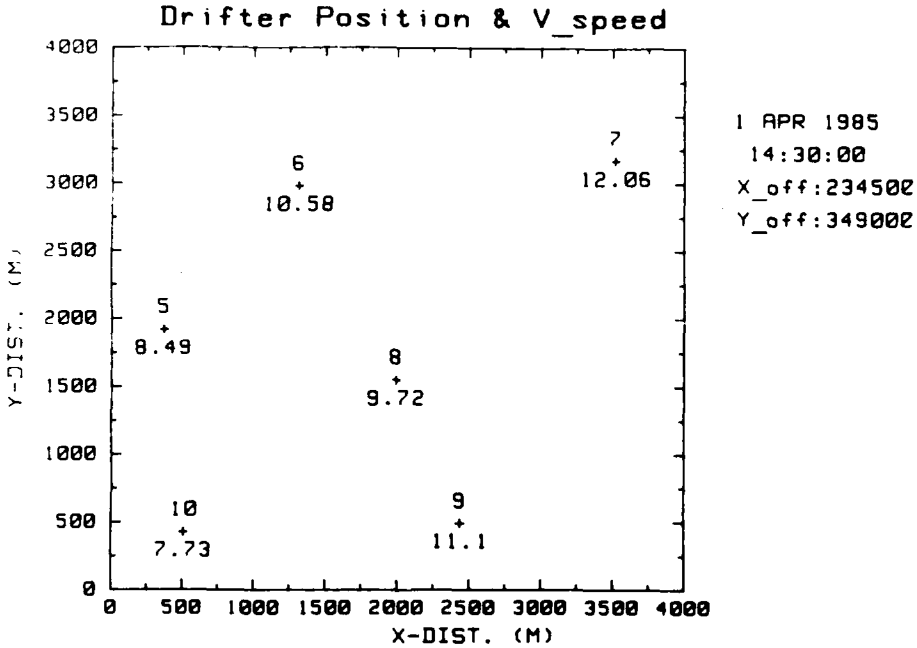


Figure 11. Position and v-component speeds at 14:30, 1 April 1985. The symbols "+" indicate the drifter location. The numerics above the "+" denote the drifter ID; while the numerics below "+" denote the values of the v-component (cm/sec).

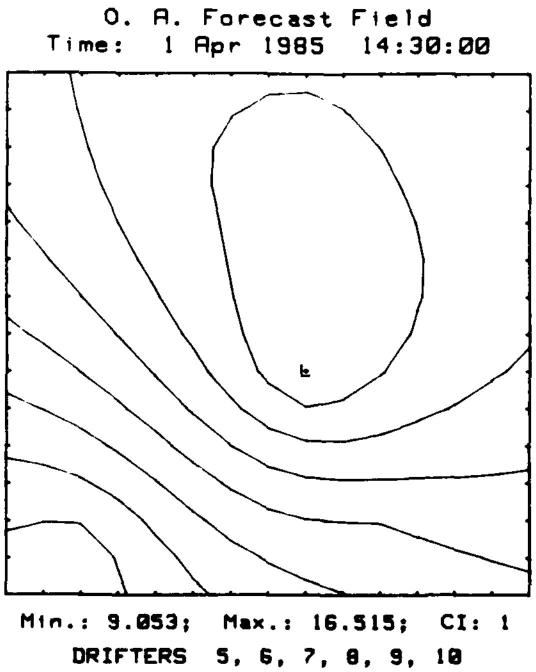


Figure 12. Objective Analysis Forecast Field of the u-component of velocity for Case 6.

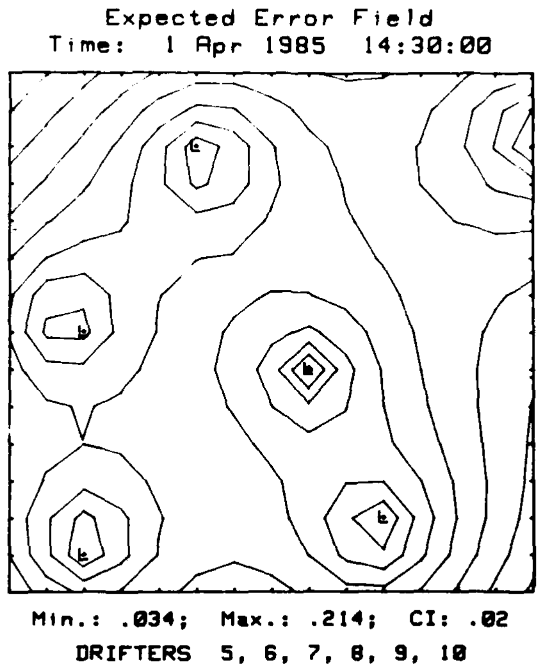


Figure 13. Expected Error Field of the u-component of velocity for Case 6.

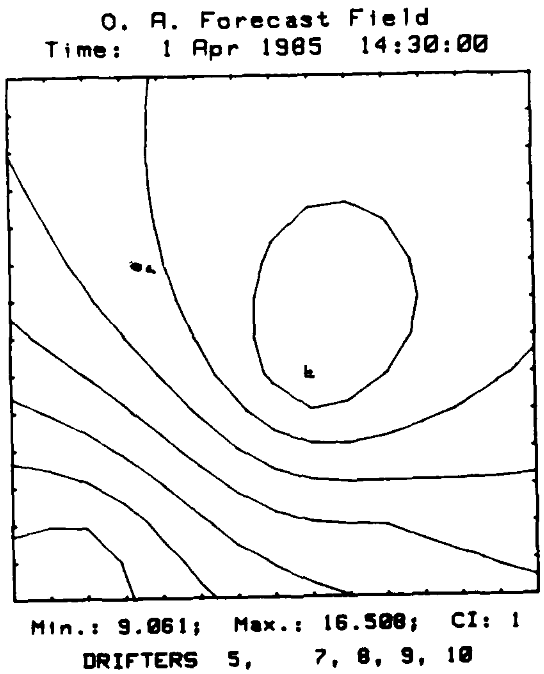


Figure 14. Objective Analysis Forecast Field of the u-component of velocity for Case 5A.

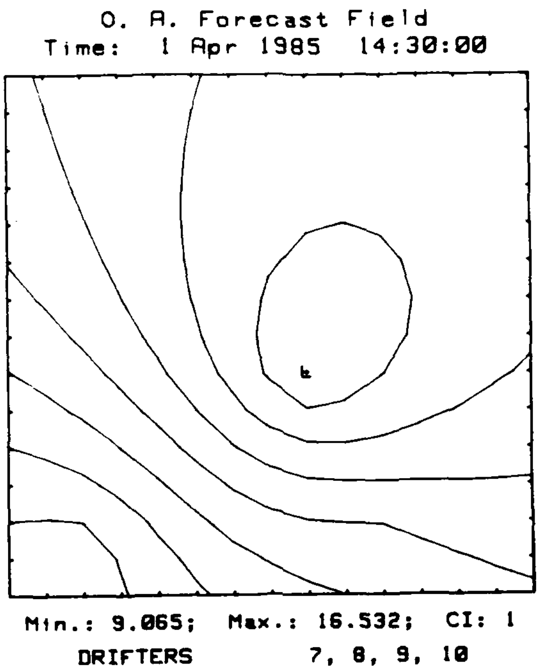


Figure 15. Objective Analysis Forecast Field of the u-component of velocity for Case 4A.

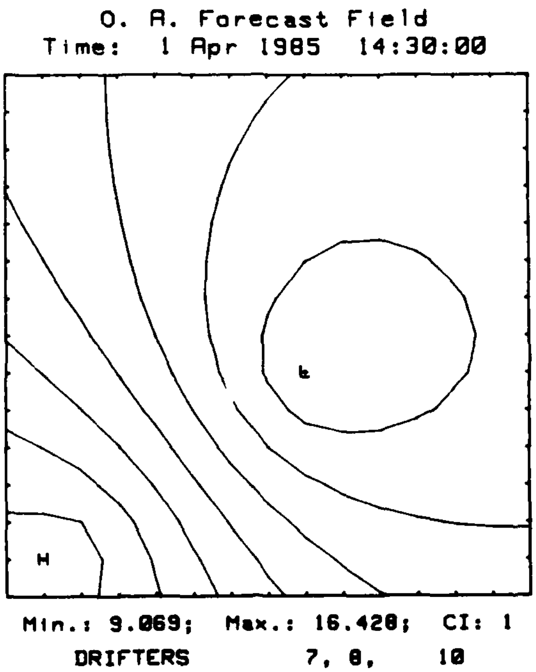


Figure 16. Objective Analysis Forecast Field of the u-component of velocity for Case 3A.

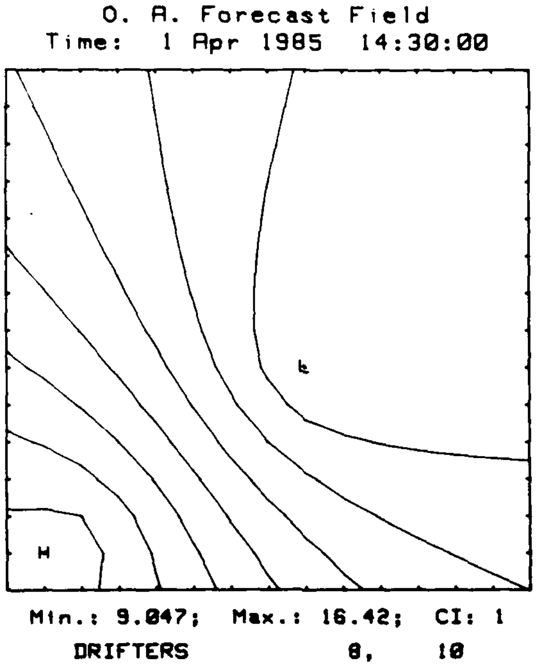


Figure 17. Objective Analysis Forecast Field of the u-component of velocity for Case 2A.

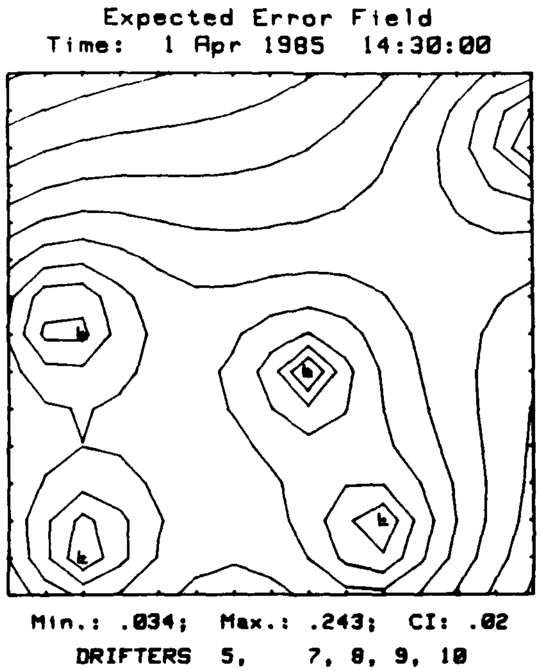


Figure 18. Expected Error Field of the u-component of velocity for Case 5A.

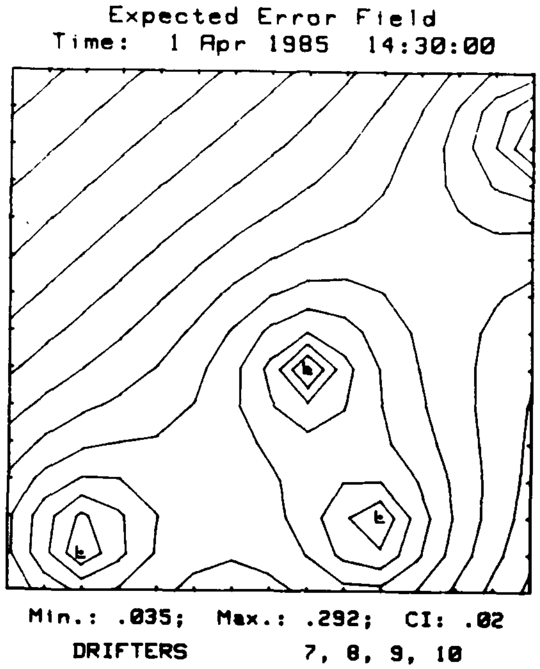


Figure 19. Expected Error Field of the u-component of velocity for Case 4A.

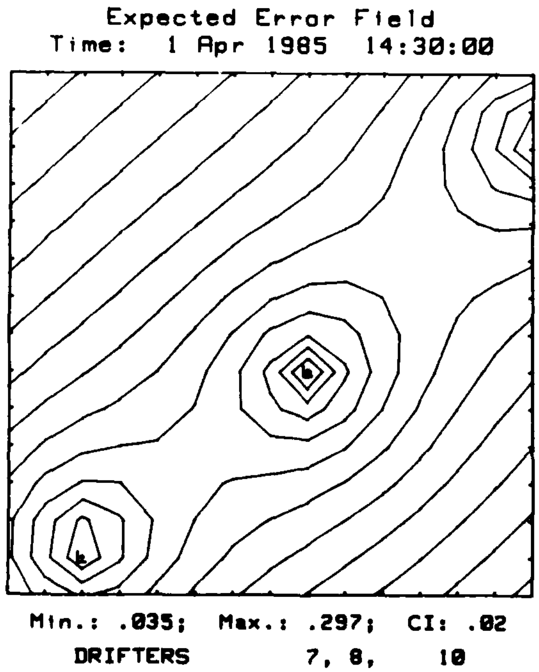
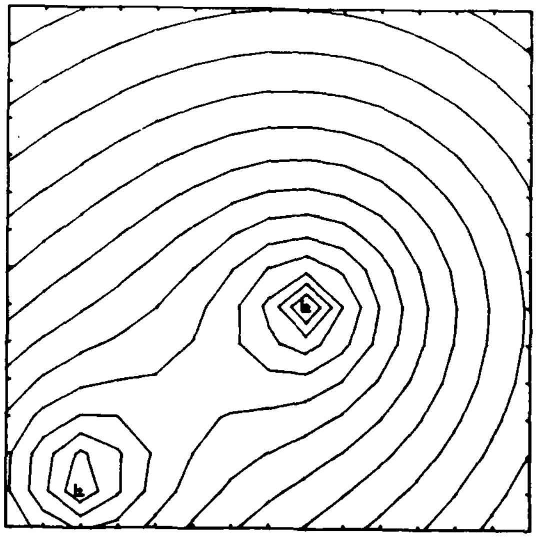


Figure 20. Expected Error Field of the u-component of velocity for Case 3A.

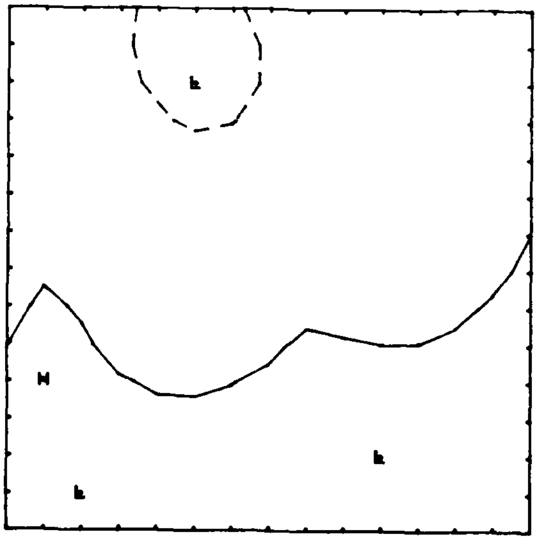
Expected Error Field
Time: 1 Apr 1985 14:30:00



Min.: .035; Max.: .313; CI: .02
DRIFTERS 8, 10

Figure 21. Expected Error Field of the u-component of velocity for Case 2A.

O. A. Forecast Field
Time: 1 Apr 1985 14:30:00



Min.: -.595; Max.: .037; CI: .5
MEASURED ERROR (6 - 5A)

Figure 22. Measured Error Field of the u-component of velocity for Case 5A.

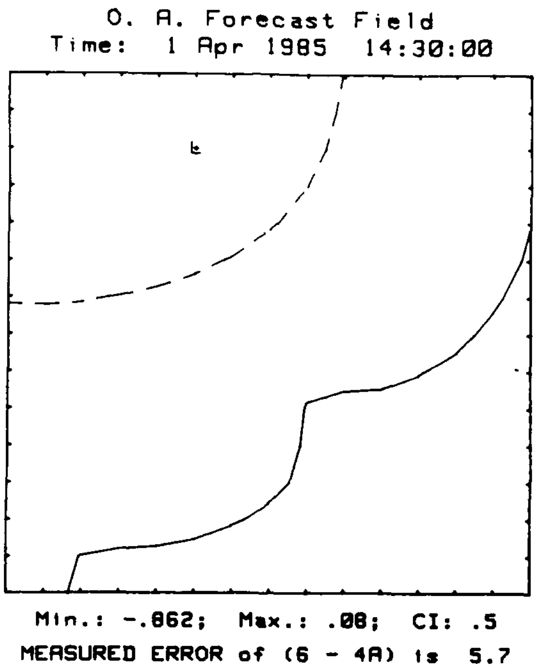


Figure 23. Measured Error Field of the u-component of velocity for Case 4A.

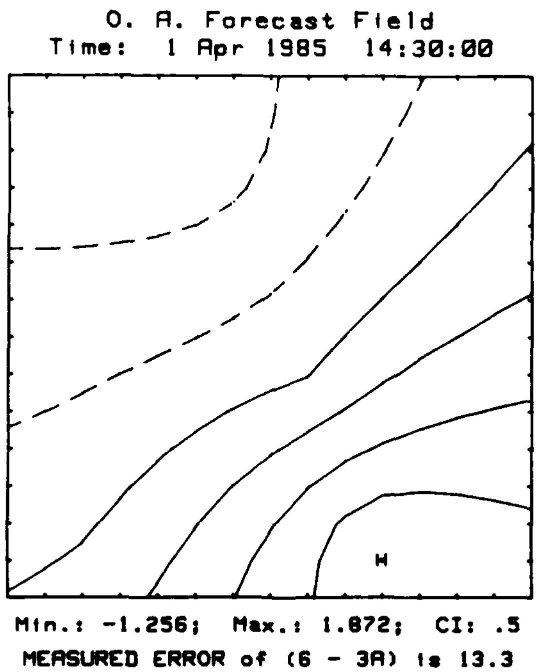


Figure 24. Measured Error Field of the u-component of velocity for Case 3A.

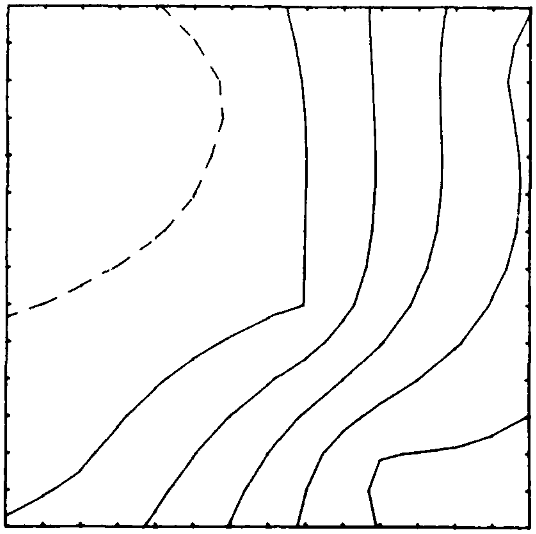


Figure 25. Measured Error Field of the u-component of velocity for Case 2A.

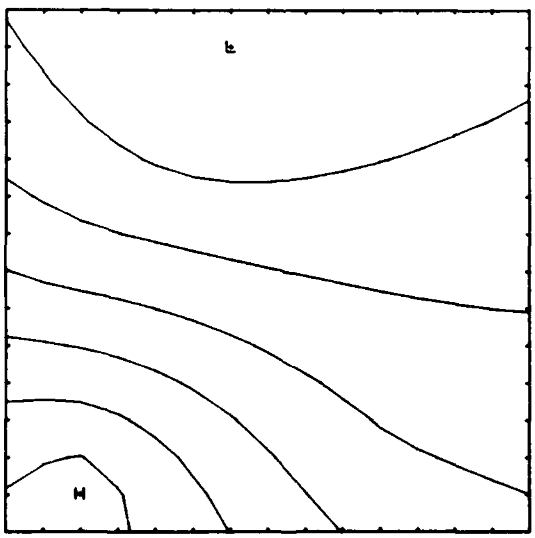
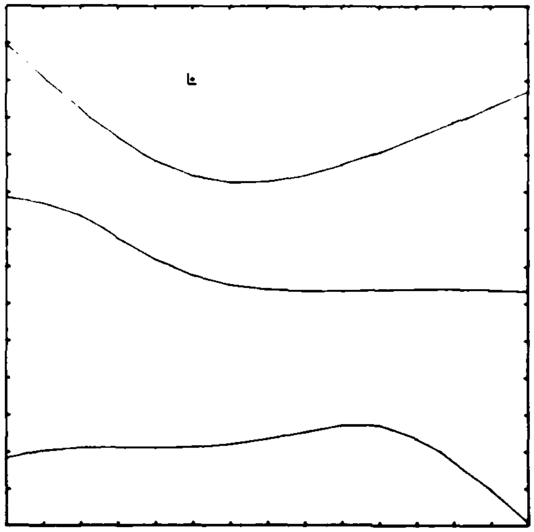


Figure 26. Objective Analysis Forecast Field of the u-component of velocity for Case 5B.

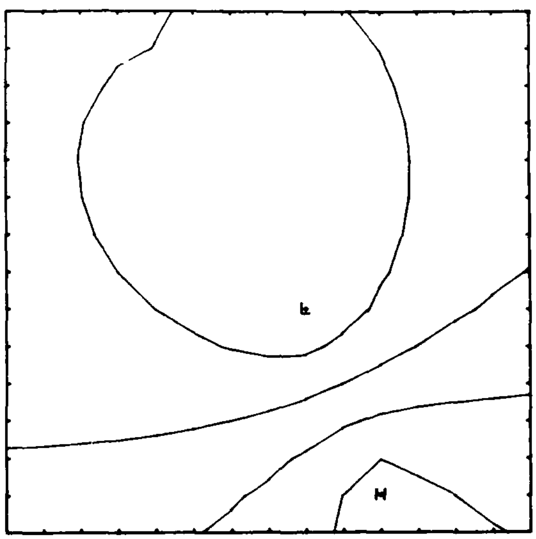
O. A. Forecast Field
Time: 1 Apr 1985 14:30:00



Min.: 10.04; Max.: 13.343; CI: 1
DRIFTERS 5, 6, 7, 9,

Figure 27. Objective Analysis Forecast Field of the u-component of velocity for Case 4B.

O. A. Forecast Field
Time: 1 Apr 1985 14:30:00



Min.: 9.054; Max.: 13.278; CI: 1
DRIFTERS 6, 8, 9

Figure 28. Objective Analysis Forecast Field of the u-component of velocity for Case 3B.

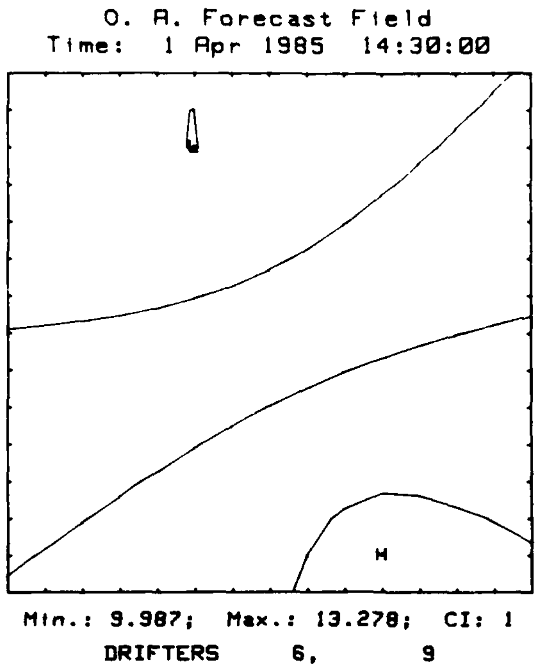


Figure 29. Objective Analysis Forecast Field of the u-component of velocity for Case 2B.

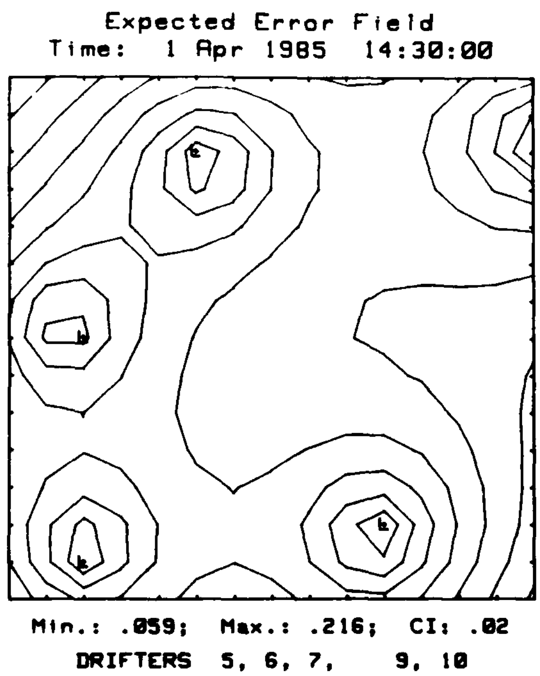


Figure 30. Expected Error Field of the u-component of velocity for Case 5B.

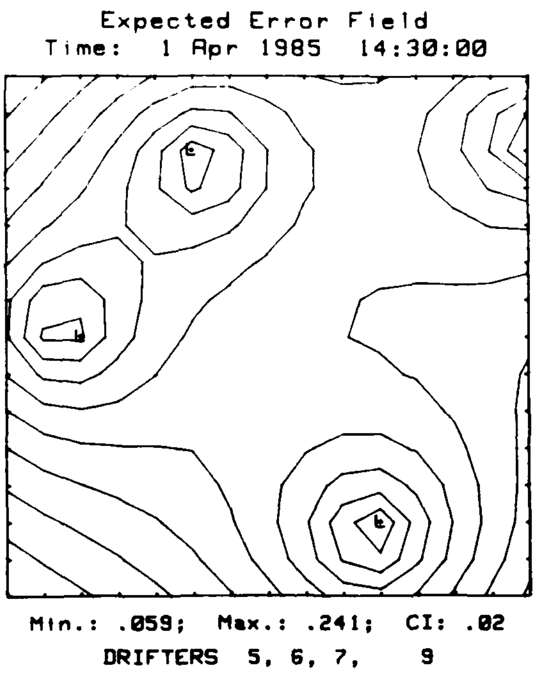


Figure 31. Expected Error Field of the u-component of velocity for Case 4B.

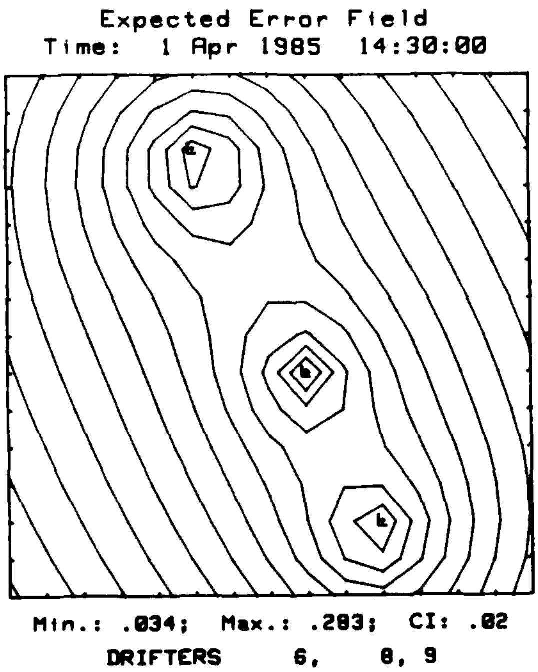


Figure 32. Expected Error Field of the u-component of velocity for Case 3B.

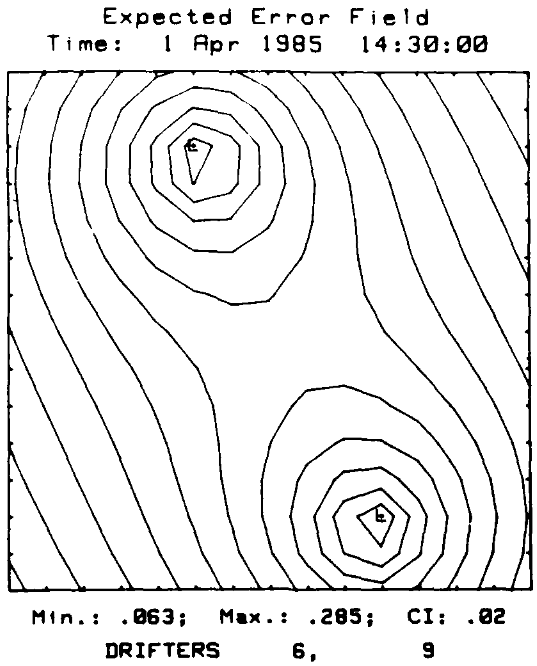


Figure 33. Expected Error Field of the u-component of velocity for Case 2B.

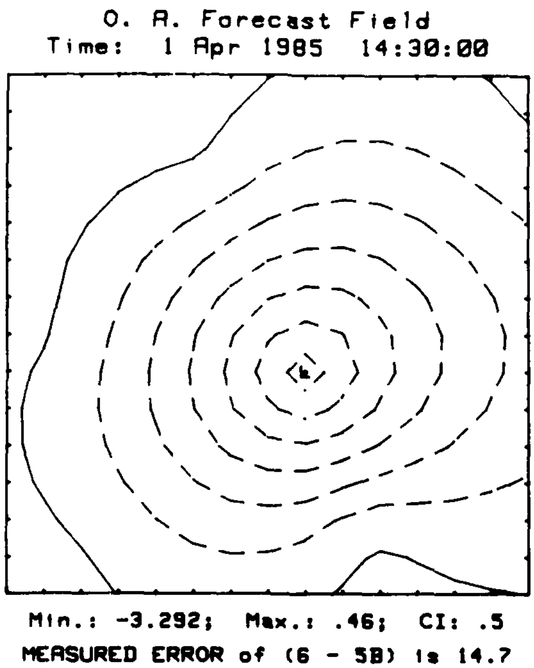
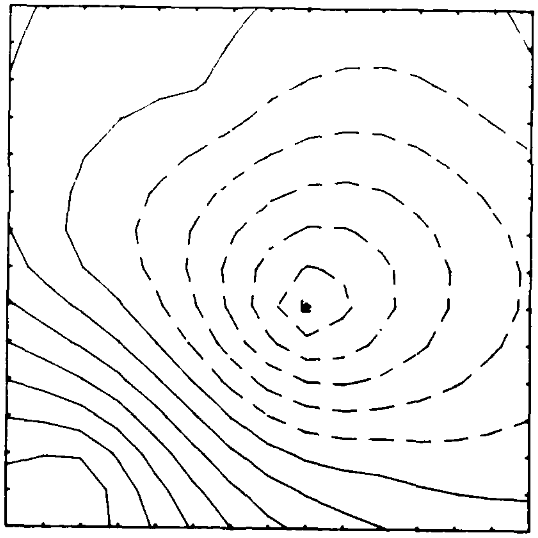


Figure 34. Measured Error Field of the u-component of velocity for Case 5B.

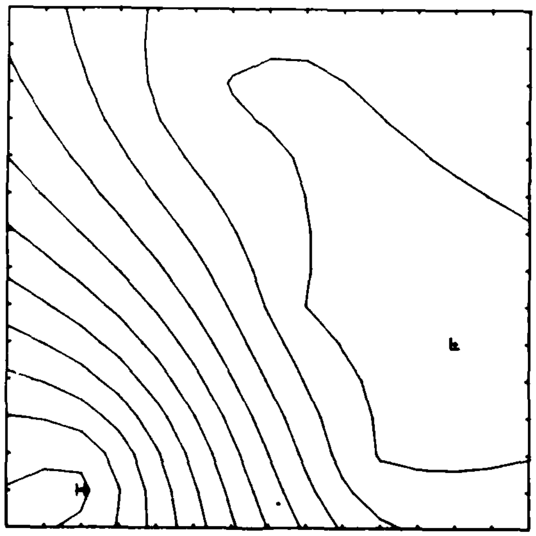
O. A. Forecast Field
Time: 1 Apr 1985 14:30:00



Min.: -3.048; Max.: 3.367; CI: .5
MEASURED ERROR of (6 - 4B) is 18.9

Figure 35. Measured Error Field of the u-component of velocity for Case 4B.

O. A. Forecast Field
Time: 1 Apr 1985 14:30:00



Min.: -.23; Max.: 5.129; CI: .5
MEASURED ERROR of (6 - 3B) is 27.3

Figure 36. Measured Error Field of the u-component of velocity for Case 3B.

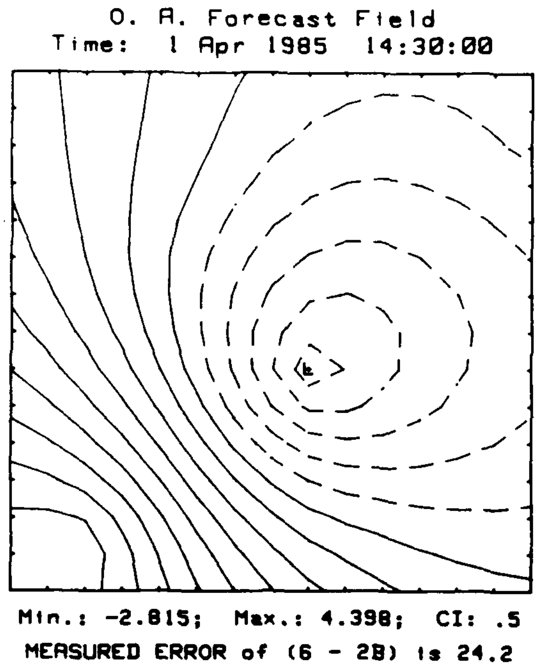


Figure 37. Measured Error Field of the u-component of velocity for Case 2B.

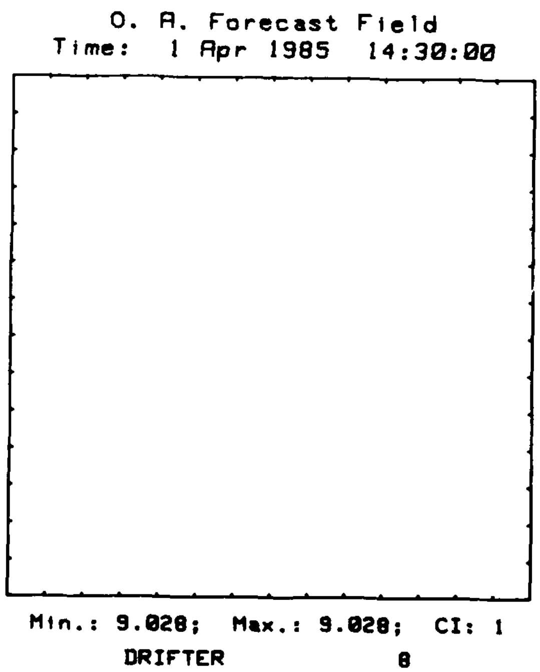


Figure 38. Objective Analysis Forecast Field of the u-component of velocity for Case 1A, Drifter No. 8.

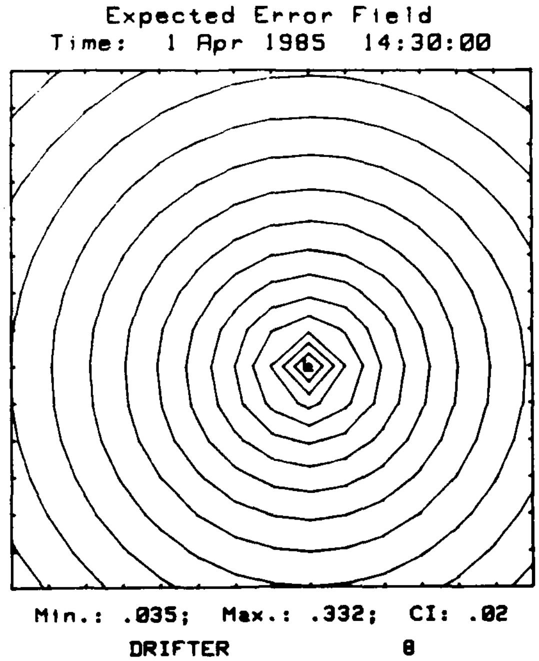


Figure 39. Expected Error Field of the u-component of velocity for Case 1A, Drifter No. 8.

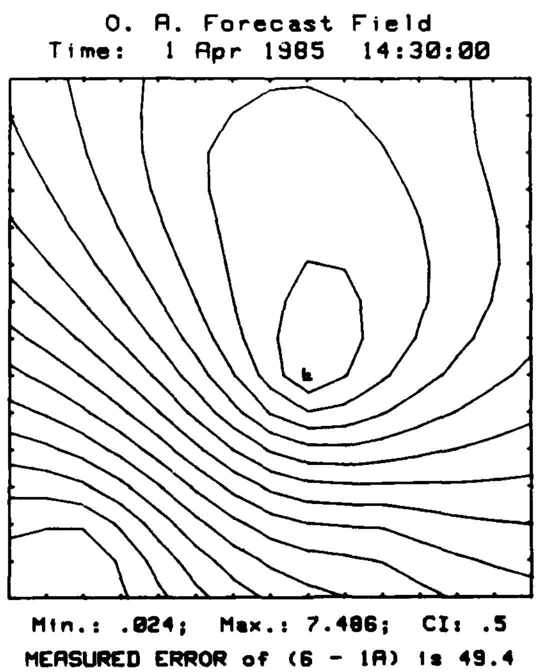


Figure 40. Measured Error Field of the u-component of velocity for Case 1A, Drifter No. 8.

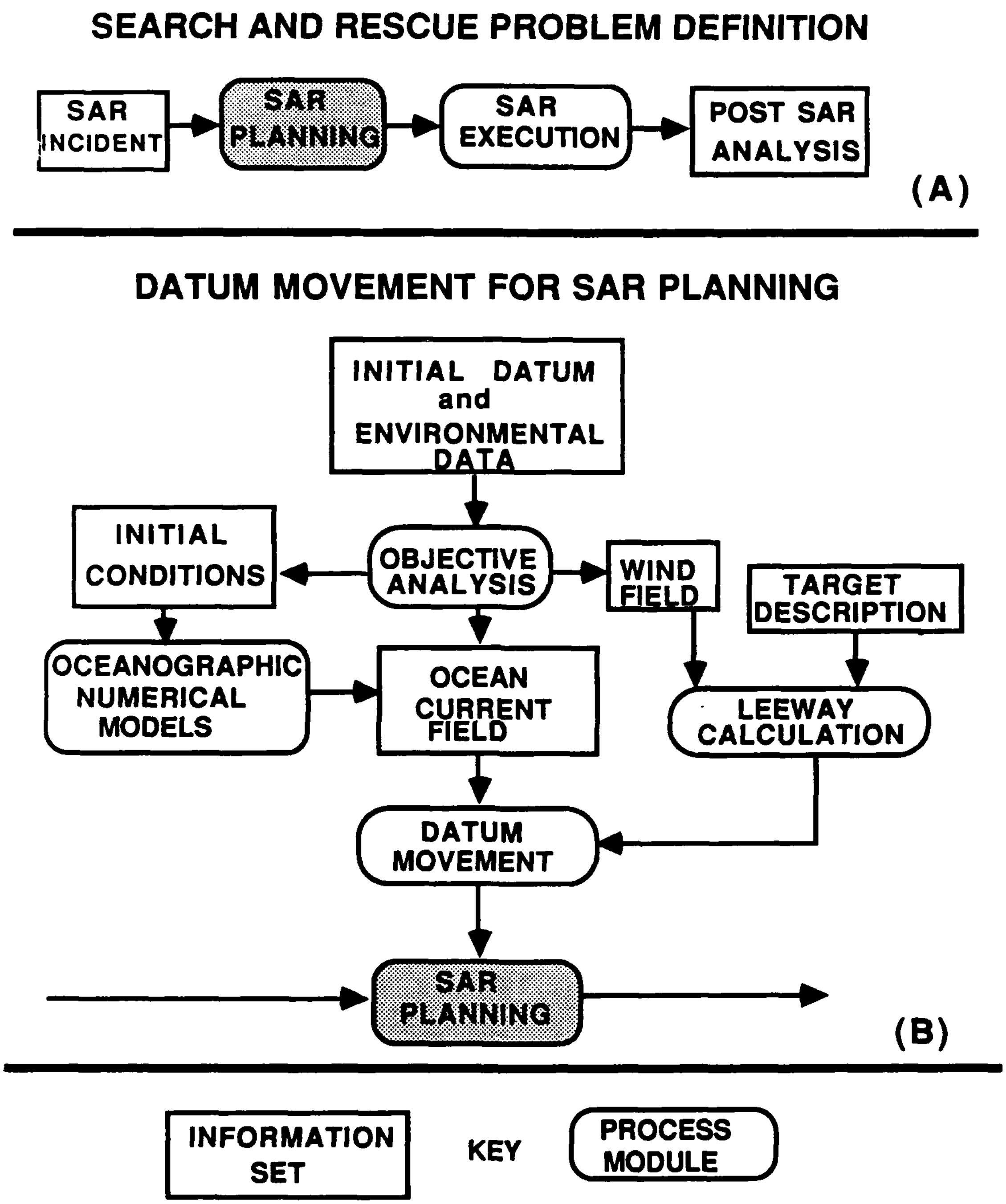


Figure 41. The Search and Rescue Problem Definition (A) and the Datum Movement for Search and Rescue Planning (B).

APPENDIX A

DEFINITIONS OF PROGRAM VARIABLES

Corr	Three-dimensional array of the correlation function.
Data_file$	Observed data file.
Date∅$, Time∅$	Date and Time of the analysis to make.
Delta_t	Time interval for each time extrapolation.
Delta_t	Time lag.
Delta_x, Delta_y	X- and y-spatial lags.
E1	The error variance of the u-component current speed.
E2	The error variance of the v-component current speed.
Es	Expected error.
Evar	Error variance.
Limit	Maximum number of influential points.
M_bin,N_bin	Center of the correlation table.
Max_space_lag	Maximum space lags.
Max_time_lag	Maximum time lags.
Ndata	Number of input observed data.
Nip	Number of inter/extrapolation positions.
Nobj	Number of objective analysis to make.
Nobs	Number of data used in the analysis.
Phiopd	Optimal data used in the analysis.
Phird	Restricted observational data.
Soa_evar$	Expected error output file for scalar objective analysis.
Soa_fcst$	Output file for scalar objective analysis forecast fields.
T_bin	Time interval in the correlation look-up table.
Tf	Forecast time.
Theta	Estimated value.
Tl	Time limit before the forecast time.
Tlimit	Maximum time radius before and after the time of the analysis.
Topd	Observation time of optimal data.
Trd	Observational time of restricted data.
Tu	Time limit after the forecast time.
UU,UV,VV	The correlation function between the current components.
Uerror,Verror	The error variance of u- and v-component current speeds.
Uest,Vest	The estimated value of u- and v-component current speeds.
Voa_evar$	Expected error output file for vector objective analysis.
Voa_fcst$	Output file for vector objective analysis forecast fields.
X_bin,Y_bin	X- and y-space intervals in the correlation look-up table.
Xip,Yip	X- and Y-coordinates of interpolation positions.

APPENDIX A (cont'd)

DEFINITIONS OF PROGRAM VARIABLES

Xlimit	Maximum spatial radius from the reference point of domain.
Xopd,Yopd	X- and Y-coordinates of optimal data.
Xrd,Yrd	X- and Y-coordinates of restricted data.

APPENDIX B

PROGRAM LISTINGS

[BLANK]

```
1000 !Program COM_COR
1005 !
1010 ! --- routine computes the correlation of observed data
1015 !     as a function  of:
1020 !     Delta_x and Delta_y (space lags)
1025 !     Delta_t (time lag).
1030 !
1035 OPTION BASE 1
1040 DIM Xdata(1089),Ydata(1089),Tdata(1089),Phi(1089)
1045 DIM Bina(33,33),Binc(33,33),Binx(33,33),Biny(33,33)
1050 !
1055 Clear$=CHR$(255)&"K"
1060 OUTPUT 2 USING "#,K";Clear$ ! clear the screen.
1065 PRINT TABXY(10,10),"Enter the number of bins in x- and y-coordinates."
1070 INPUT "Npx?,Npy?",Npx,Npy
1075 OUTPUT 2 USING "#,K";Clear$ ! clear the screen.
1080 PRINT TABXY(10,10),"Enter x- and y-bin sizes and time lag:"
1085 INPUT "X_bin?, Y_bin?, T_bin?",X_bin,Y_bin,T_bin
1090 OUTPUT 2 USING "#,K";Clear$ ! clear the screen.
1095 PRINT TABXY(10,10),"Enter number of iterations"
1100 INPUT "Iterations?",Interations
1105 OUTPUT 2 USING "#,K";Clear$ ! clear the screen.
1110 PRINT TABXY(10,10),"Enter the correlation filename"
1115 INPUT "Corr_file$?",Corr_file$
1120 OUTPUT 2 USING "#,K";Clear$ ! clear the screen.
1125 PRINT TABXY(10,10),"Enter the observed data file"
1130 INPUT "Data_file$?",Data_file$
1135 OUTPUT 2 USING "#,K";Clear$ ! clear the screen.
1140 !
1145 ! --- get data
1150 !
1155 CALL Get_data(Data_file$,Xdata(*),Ydata(*),Tdata(*),Phi(*),Ndata)
1160 Mean=0.
1165 FOR I=1 TO Ndata
1170  Mean=Mean+Phi(I)
1175 NEXT I
1180 Mean=Mean/Ndata
1185 PRINT "Number and mean of input data: ",Ndata,Mean
1190 !
1195 ! --- remove the mean of input data
1200 !
1205 FOR I=1 TO Ndata
1210  Phi(I)=Phi(I)-Mean
1215 NEXT I
1220 !
1225 Irep=0
1230 T_inc=0.
1235 M_bin=(Npx+1)/2
1240 N_bin=(Npy+1)/2
1245 REPEAT
```

```
1250   T_inc=T_inc+Tbin*Irep
1255   Irep=Irep+1
1260   FOR I=1 TO Npx
1265    FOR J=1 TO Npy
1270     Bina(I,J)=0.
1275     Binc(I,J)=0.
1280     Binx(I,J)=0.
1285     Biny(I,J)=0.
1290    NEXT J
1295   NEXT I
1300  !
1305  ! --- loop to find pairs of data separated by T_bin+-5 time units.
1310  !
1315   FOR J=1 TO Ndata
1320    FOR I=1 TO Ndata
1325     Delta_t=Tdata(J)-Tdata(I)
1330     IF Delta_t>T_inc+.5*T_bin THEN GOTO Next_i
1335     IF Delta_t<T_inc-.5*T_bin THEN GOTO Next_i
1340     Delta_x=Xdata(J)-Xdata(I)
1345     Delta_y=Ydata(J)-Ydata(I)
1350     Idx=INT((Delta_x+.5*X_bin)/X_bin)+M_bin
1355     Idy=INT((Delta_y+.5*Y_bin)/Y_bin)+N_bin
1360     IF Idx<1 OR Idx>Npx THEN GOTO Next_i
1365     IF Idy<1 OR Idy>Npy THEN GOTO Next_i
1370     Binc(Idx,Idy)=Binc(Idx,Idy)+1
1375     Bina(Idx,Idy)=Phi(J)*Phi(I)+Bina(Idx,Idy)
1380     Binx(Idx,Idy)=Phi(J)*Phi(J)+Binx(Idx,Idy)
1385     Biny(Idx,Idy)=Phi(I)*Phi(I)+Biny(Idx,Idy)
1390 Next_i:NEXT I
1395   NEXT J
1400  !
1405  ! --- normalize bin
1410  !
1415   BEEP
1420   FOR J=1 TO Npy
1425    FOR I=1 TO Npx
1430     IF Binx(I,J)=0. OR Biny(I,J)=0. THEN GOTO Nexti
1435     Bina(I,J)=Bina(I,J)/SQR(Binx(I,J)*Biny(I,J))
1440 Nexti: NEXT I
1445   NEXT J
1450  !
1455  ! --- output correlation matrix
1460  !
1465   ASSIGN @Path_out TO Corr_file$
1470   FOR I=1 TO Npx
1475    FOR J=1 TO Npy
1480     OUTPUT @Path_out;Bina(I,J)
1485    NEXT J
1490   NEXT I
1495  UNTIL Irep=Interations
```

```
1500 Finish:  !
1505  ASSIGN @Path_out TO *
1510  END
1515  !
1520  !
1525  !
1530 Get_data:SUB Get_data(Filename$,Xdata(*),Ydata(*),Tdata(*),Phidata(*),Ndata
)
1535   ASSIGN @Path_in TO Filename$
1540 Read_in_data:   !
1545   Ndata=0
1550   PRINT "      **********   Echo check of the first ten records    **********
*"
1555   PRINT "  No.     X_POS           Y_POS                TIME             DATA"
1560   ON END @Path_in GOTO 1610
1565   DISP USING "K,K,K";"Reading the observed data from ",Filename$,", please
wait."
1570   ENTER @Path_in;X,Y,T,Phi,Dummy
1575   IF Ndata<=10 THEN PRINT USING "DDDD,XX,MD.DDDDDDE,XX,MD.DDDDDDE,XX,MD.DDD
DDDDDE,XX,MD.DDDDDDE";Ndata,X,Y,T,Phi
1580   Ndata=Ndata+1
1585   Xdata(Ndata)=X
1590   Ydata(Ndata)=Y
1595   Tdata(Ndata)=T
1600   Phidata(Ndata)=Phi
1605   GOTO 1560
1610   ASSIGN @Path_in TO *! Closing input_file.
1615   DISP
1620  SUBEND
```

```
1000  !
1005  ! SUBPROGRAM FNCov_f
1010  !
1015  DEF FNCov(Delx,Dely,Delt,Xfold,Yfold,Tfold)
1020  !
1025  ! routine gets the correlation function from a fitted formula.
1030  !
1035    OPTION BASE 1
1040    RAD
1045    COM /Efield/ Evar
1046    RETURN EXP(-SQR((Delx/Xfold)*(Delx/Xfold)+(Dely/Yfold)*(Dely/Yfold)+(Delt
/Tfold)*(Delt/Tfold)))
1050  !   R2=SQR((Delx^2+Dely^2)/5.0E+4^2)
1055  !   Cor=EXP(-R2)
1060  !   RETURN Cor
1065  FNEND
```

```
1000  ! SUBPROGRAM FNCov_t
1005  !
1010  DEF FNCov(Delta_x,Delta_y,Delta_t)
1015  !
1020  ! --- routine determines the correlation function from the look-up table,
1025  !     given space and time lags.
1030  !
1035  !     Corr is a three-dimensional array of the correlation function.
1040  !     Delta_x and Delta_y are the x- and y-spatial lags.
1045  !     Delta_t is the time lag.
1050  !     M_bin is the mid-point in the x-dimension of correlation table.
1055  !     N_bin is the mid-point in the y-dimension of correlation table.
1060  !     T_bin is the time interval in the correlation look-up table.
1065  !     X_bin is the x-distance interval in the correlation look-up table.
1070  !     Y_bin is the y-distance interval in the correlation look-up table.
1075   OPTION BASE 1
1080   RAD
1085   COM /Corr/ Corr(33,33,10),X_bin,Y_bin,T_bin,M_bin,N_bin
1090   Idx=INT((Delta_x+.5*X_bin)/X_bin)+M_bin
1095   Idy=INT((Delta_y+.5*Y_bin)/Y_bin)+N_bin
1100   Idt=INT((Delta_t+.5*T_bin)/T_bin)+1
1105   Cor=Corr(Idx,Idy,Idt)
1110   RETURN Cor
1115  FNEND
```

```
1000  !PROGRAM GET_DATA
1005  !
1010  DIM X(1089),Y(1089),T(1089),U(1089),V(1089)
1015  ! DATA "DRIFTER_5:,700,1",0
1020  ! DATA "DRIFTER_6:,700,1",0
1025  DATA "DRIFTER_7:,700,1",0
1030  DATA "DRIFTER_8:,700,1",0
1035  DATA "DRIFTER_9:,700,1",0
1040  DATA "DRIFTER_10:,700,1",0
1041  ON ERROR GOTO 1046
1042  CREATE BDAT "OBSDAT4A:,700,1",100
1043  CREATE BDAT "SOA_FCST4A:,700,1",10
1044  CREATE BDAT "SOA_EVAR4A:,700,1",10
1046  OFF ERROR
1047  ASSIGN @Pathout TO "OBSDAT4A:,700,1"
1050  FOR N=1 TO 4
1055   READ Filename$,Nskip
1060   PRINT Filename$
1065   Get_data(Filename$,Nskip,X(*),Y(*),T(*),U(*),V(*),Ndata)
1070   PRINT Ndata
1075   FOR I=1 TO Ndata
1080    OUTPUT @Pathout;X(I),Y(I),T(I),U(I),V(I)
1085   NEXT I
1090  NEXT N
1095  END
1100 Get_data:SUB Get_data(Filename$,Nskip,Xdata(*),Ydata(*),Tdata(*),Udata(*),V
data(*),Ndata)
1105   DIM Drno$[2],Id$[1],Desc$[30]
1110   INTEGER Npos
1115   ASSIGN @Path_in TO Filename$
1120   ENTER @Path_in;Drno$,Id$,T$,Npos
1125   Ndata=0
1130   ON END @Path_in GOTO 1180
1135   ENTER @Path_in;X,Y,T,U,V
1140   IF Ndata<10 THEN PRINT X,Y,T,U
1145   Ndata=Ndata+1
1150   Xdata(Ndata)=X
1155   Ydata(Ndata)=Y
1160   Tdata(Ndata)=T
1165   Udata(Ndata)=U
1170   Vdata(Ndata)=V
1175   GOTO 1130
1180   ASSIGN @Path_in TO *! Closing input_file.
1185  SUBEND
```

```
1000  ! Program LINDAT UV
1005  !
1010  !      Routine plots the u  and v component current speeds as a
1015  !      function of time. The solid and dotted lines indicate the
1020  !      u  and v component speeds respectively.
1025  !
1030  OPTION BASE 1
1035  DIM Udata(1089),Vdata(1089),T(1089)
1040  ALLOCATE A$[10]
1045  !Title of this plot
1050  DATA "DRIFTER 5"
1055  ! X axis Label
1060  DATA "1 APR 1985"
1065  ! Y axis label
1070  DATA "cm/sec"
1075  !     X LEFT,X RIGHT,X tick spacing,X major count
1080  DATA       1,     241,             5,             6
1085  !     Y bottom,  Y top,Y tick spacing,Y major count
1090  DATA        0.,     25.,             1,             5
1095  !     Dump graphics to printer (yes/no; 1/0)
1100  DATA 0
1105  ! -- Data sampling interval(seconds)
1110  DATA 120.
1115  ! --- the origin of time axis
1120  DATA "1 APR 1985"
1125  DATA "10:00:00"
1130  !input file name, number of data to be skipped
1135  DATA "DRIFTER 5:,700,1",0
1140  C$=CHR$(255)&"K"
1145  READ Title$
1150  READ X label$
1155  READ Y_label$
1160  ON KBD GOTO Exit
1165  READ X left,X right,X tick_spacing,X_major count
1170  READ Y bottom,Y top,Y tick spacing,Y_major count
1175  READ Dump graphics
1180  Major tick size=3.0
1185  OUTPUT 2 USING "#,K";C$        ! Clear screen for graph
1190  READ Delta t
1195  READ Date0 $
1200  READ Time0_$
1205  Time0=DATE(Date0 $)+TIME(Time0 $)
1210  READ Input file$,Nskip
1215  Get data(Input file$,Nskip,Udata(*),Vdata(*),T(*),Ndata)  ! Get data for p
lotting
1220  OUTPUT 2 USING "#,K";C$        ! Clear screen for graph
1225  GINIT                           ! Initialize various graphics parameters.
1230 !PLOTTER IS 3,"INTERNAL"         ! Use the internal screen
1231  PLOTTER IS 705,"HPGL  "         ! Use the external HPGL plotter at HP-IB add
ress 705
```

```
1235  GRAPHICS ON                      ! Turn on the graphics screen
1240  LORG 6                           ! Reference point: center of top of label
1245  ! Determine how many GDUs wide and high the screen is
1250  Gdu(X_gdu_max,Y_gdu_max)
1255  FOR I=-.2 TO .2 STEP .1          ! Offset of X from starting point
1260   MOVE X_gdu_max/2+I,Y_gdu_max    ! Move to about middle of top of screen
1265   LABEL USING "#,K";Title$        ! Write title of plot
1270  NEXT I                           ! Next position for title
1275  DEG                              ! Angular mode is degrees (used in LDIR)
1280  Label(4.8,.6,90,6,1,0.,Y_gdu_max/2,Y_label$)  ! Write Y_axis label
1285  Label(3.8,.6,0,4,1,X_gdu_max/2,.01*Y_gdu_max,X_label$)! Write X_axis label
1290  ! Define subset of screen area
1295  VIEWPORT .1*X_gdu_max,.9*X_gdu_max,.1*Y_gdu_max,.9*Y_gdu_max
1300  ! Anisotropic scaling: left/right/bottom/top
1305  WINDOW X_left,X_right,Y_bottom,Y_top*1.01
1310  ! Draw axes intersecting at lower left
1315  Y_axis_location=X_left
1320  X_axis_location=Y_bottom
1325  AXES X_tick_spacing,Y_tick_spacing,Y_axis_location,X_axis_location,X_major
_count,Y_major_count,Major_tick_size
1330  CLIP OFF                         ! So labels can be outside VIEWPORT limits
1335  CSIZE 3.5,.6                     ! Smaller chars for axis labelling
1340  LORG 6                           ! Ref. pt: Top center    !\
1345  WINDOW X_left,X_right,.1*Y_gdu_max,.9*Y_gdu_max
1350  X_step=X_tick_spacing*X_major_count
1355  FOR I=X_left TO X_right STEP X_step !                      ! \
1360   MOVE I-X_left+1,.09*Y_gdu_max!A smidgeon below X-axis !  >  Label X-axis
1365   A$=TIME$(Time0+(I-1)*Delta_t)
1370   LABEL USING "#,K";A$[1;2]
1375  NEXT I
1380  WINDOW X_left,X_right,Y_bottom,Y_top*1.01
1385  LORG 8                           ! Ref. pt: Right center     !\
1390  Y_step=Y_tick_spacing*Y_major_count
1395  FOR I=Y_bottom TO Y_top STEP Y_step !                         ! \
1400   MOVE .8,I                       ! Smidgeon left of Y-axis   !  > Label Y-axis
1405   LABEL USING "#,K";I             ! DD.D; no CR/LF            ! /
1410  NEXT I                           ! et sequens                !/
1415  PENUP                            ! LABEL statement leaves the pen down
1420 ! Anisotropic scaling: left/right/bottom/top
1425  WINDOW X_left,X_right,Y_bottom,Y_top*1.01
1430  Is=INT(T(1)-Time0)/Delta_t+1
1435  LINE TYPE 1
1440  FOR I=1 TO Ndata                 ! Points to be plotted...
1445   PLOT I+Is,Udata(I)              ! Get a data point and plot it against X
1450  NEXT I
1455  PENUP
1460  LINE TYPE 8
1465  FOR I=1 TO Ndata                  ! Points to be plotted...
1470   PLOT I+Is,Vdata(I)               ! Get a data point and plot it against X
1475  NEXT I
```

```
1480 !  DISP "Enter 'Space bar' to go on"
1485  WAIT 1
1490  DISP
1495  GOTO 1480                       ! View the plot as long as you want
1500 Exit:IF Dump_graphics THEN
1505 ! DUMP GRAPHICS CRT TO #701
1510  END IF
1515  GRAPHICS OFF
1520  OUTPUT 2 USING "#,K";C$
1525  END                             ! finis
1530  !
1535 Gdu:SUB Gdu(X_gdu_max,Y_gdu_max,OPTIONAL Gdu_xmid,Gdu_ymid)
1540  ! This returns Xright, Yhigh and their respective midpoints in GDUs.
1545  ! Note that if Gud_xmid is defined, Gdu_ymid must be also.
1550   COM /G_units/ Gdu_xmax,Gdu_ymax,Udu_xmin,Udu_xmax,Udu_ymin,Udu_ymax,Show
1555   IF Gdu_xmax=0 THEN
1560    Gdu_xmax=100*MAX(1,RATIO)
1565    Gdu_ymax=100*MAX(1,1/RATIO)
1570   END IF
1575   X_gdu_max=Gdu_xmax
1580   Y_gdu_max=Gdu_ymax
1585   IF NPAR>2 THEN
1590    Gud_xmid=Gdu_xmax*.5
1595    Gud_ymid=Gdu_ymax*.5
1600   END IF
1605  SUBEND
1610 Label:SUB Label(Csize,Asp_ratio,Ldir,Lorg,Pen,X,Y,Text$)
1615  ! This defines several systems variables (in CSIZE, LDIR, etc.), and
1620  ! labels the test (if any) accordingly.
1625   DEG
1630   CSIZE Csize,Asp_ratio
1635   LDIR Ldir
1640   LORG Lorg
1645   PEN Pen
1650   MOVE X,Y
1655   IF Text$<>"" THEN LABEL USING "#,K";Text$
1660   PENUP
1665  SUBEND
1670  !
1675 Get_data:SUB Get_data(File_in$,Nskip,Udata(*),Vdata(*),T(*),Ndata)
1680  !
1685   OPTION BASE 1
1690   DIM Drno$[2],Id$[1],Desc$[30]
1695   INTEGER Npos
1700   ASSIGN @Path_in TO File_in$
1705  !read in the header
1710   ENTER @Path_in;Drno$,Id$,Desc$,Npos
1715   IF Nskip=0 THEN GOTO Read_in_data
1720   FOR N=1 TO Nskip
1725    ENTER @Path_in;Xd,Yd,Td,Ud,Vd
```

```
1730   NEXT N
1735 Read_in_data:I=1
1740   ON END @Path_in GOTO 1760
1745   ENTER @Path_in;X,Y,T(I),Udata(I),Vdata(I)
1750   I=I+1
1755   GOTO 1740
1760   Ndata=I-1
1765   ASSIGN @Path_in TO *
1770  SUBEND
```

```
1000  ! PROGRAM: PLOT_PUS
1005  !
1010  ! --- Routine plots drifter's position and u-component speed.
1015  !
1020  ! ----------------- INPUT DATA SECTION --------------------------------
1025  ! Title$: Title of this plot
1030  DATA "Drifter Position & U_speed"
1035  ! ---------------------------------------------------------------------
1040  ! X_label$: X_axis Label
1045  DATA "X-DIST. (M)"
1050  ! ----------------------------------------------------------------------
1055  ! Y_label$: Y_axis label
1060  DATA "Y-DIST. (M)"
1065  ! ----------------------------------------------------------------------
1070  !    X_left,X_right,X_tick_spacing,X_major_count
1075  DATA       0,   4000,            250,            2
1080  ! ----------------------------------------------------------------------
1085  !    Y_bottom,  Y_top,Y_tick_spacing,Y_major_count
1090  DATA       0,   4000,            250,            2
1095  ! -----------------------------------------------------------------------
1100  ! Grid: Need grid lines (yes/no; 1/0)
1105  DATA 0
1110  ! -----------------------------------------------------------------------
1115  ! Dump_graphics: Dump graphics to printer (yes/no; 1/0)
1120  DATA 1
1125  ! -----------------------------------------------------------------------
1130  ! Xoff,Yoff; offset in the x- and y- directions respectively.
1135  DATA 234500,349000
1140  ! -----------------------------------------------------------------------
1145  ! Date_$: plotting date
1150  DATA 1 APR 1985
1155  ! -----------------------------------------------------------------------
1160  ! Time_$: plotting time
1165  DATA 14:30:00
1170  ! -----------------------------------------------------------------------
1175  ! Ndrifter: number of drifters to be plotted, and titles of the drifters
1180  DATA 6, 5, 6, 7, 8, 9, 10
1185  ! -----------------------------------------------------------------------
1190  ! Input_file$: input file name
1195  DATA "DRIFTER_5:,700,1"
1196  DATA "DRIFTER_6:,700,1"
1200  DATA "DRIFTER_7:,700,1"
1205  DATA "DRIFTER_8:,700,1"
1210  DATA "DRIFTER_9:,700,1"
1211  DATA "DRIFTER_10:,700,1"
1215  !
1220  ! --------------------- END OF INPUT DATA -------------------------
1225  !
1230  OPTION BASE 1
1235  DIM Title$[30],Id$[2]
```

```
1240  DIM Date_$[11],Time_$[8]
1245  READ Title$
1250  READ X_label$
1255  READ Y_label$
1260  READ X_left,X_right,X_tick_spacing,X_major_count
1265  READ Y_bottom,Y_top,Y_tick_spacing,Y_major_count
1270  READ Grid
1275  READ Dump_graphics
1280  READ Xoff,Yoff
1285  READ Date_$
1290  READ Time_$
1295  Time=DATE(Date_$)+TIME(Time_$)
1300  Major_tick_size=3.0
1305  C$=CHR$(255)&"K"
1310  OUTPUT 2 USING "#,K";C$           ! Clear screen for graph
1315  GINIT                            ! Initialize various graphics parameters.
1320 ! PLOTTER IS 3,"INTERNAL"          ! Use the internal screen
1321   PLOTTER IS 705,"HPGL"           ! Use the external HPGL pen plotter at HP-IP
 Address 705
1325  GRAPHICS ON                      ! Turn on the graphics screen
1330  LORG 6                           ! Reference point: center of top of label
1335  ! Determine how many GDUs wide and high the screen is
1340  Gdu(X_gdu_max,Y_gdu_max)
1345  FOR I=-.1 TO .1 STEP .1          ! Offset of X from starting point
1350   MOVE X_gdu_max/2.3+I,Y_gdu_max ! Move to about middle of top of screen
1355   LABEL USING "#,K";Title$        ! Write title of plot
1360  NEXT I                           ! Next position for title
1365  DEG                              ! Angular mode is degrees (used in LDIR)
1370  Label(3.8,.6,90,6,1,0.,Y_gdu_max/2,Y_label$)  ! Write Y_axis label
1375  Label(3.8,.6,0,4,1,X_gdu_max/2,.01*Y_gdu_max,X_label$) ! Write X_axis labe
l
1380  Label(3.8,.6,0,4,1,.85*X_gdu_max,.8*Y_gdu_max,Date_$)
1385  Label(3.8,.6,0,4,1,.85*X_gdu_max,.75*Y_gdu_max,Time_$)
1390  Label(3.8,.6,0,4,1,.87*X_gdu_max,.7*Y_gdu_max,"X_off:"&VAL$(Xoff))
1395  Label(3.8,.6,0,4,1,.87*X_gdu_max,.65*Y_gdu_max,"Y_off:"&VAL$(Yoff))
1400  ! Define subset of screen area
1405  VIEWPORT .12*X_gdu_max,.75*X_gdu_max,.1*Y_gdu_max,.93*Y_gdu_max
1410 ! Anisotropic scaling: left/right/bottom/top
1415  WINDOW X_left,X_right,Y_bottom,Y_top
1420  ! Draw a box
1425  AXES X_tick_spacing,Y_tick_spacing,X_left,Y_bottom,X_major_count,Y_major_c
ount,Major_tick_size
1430  AXES X_tick_spacing,Y_tick_spacing,X_right,Y_top,X_major_count,Y_major_cou
nt,Major_tick_size
1435  IF Grid THEN GRID X_tick_spacing,Y_tick_spacing,X_left,Y_bottom,X_major_co
unt,Y_major_count
1440  CLIP OFF                         ! So labels can be outside VIEWPORT limits
1445  CSIZE 3.5,.6                     ! Smaller chars for axis labelling
1450  LORG 6                           ! Ref. pt: Top center      !\
1455  WINDOW X_left,X_right,.1*Y_gdu_max,.9*Y_gdu_max
```

```
1460  X_step=X_tick_spacing*X_major_count
1465  FOR I=X_left TO X_right STEP X_step! Every X_STEP units ! \
1470   MOVE I-X_left+1,.09*Y_gdu_max! A smidgeon below X-axis !  >  Label X-axis
1475   LABEL USING "#,K";I          !                           ! /
1480  NEXT I                        !                           !/
1485  WINDOW X_left,X_right,Y_bottom,Y_top
1490  LORG 8                        ! Ref. pt: Right center    !\
1495  Y_step=Y_tick_spacing*Y_major_count
1500  FOR I=Y_bottom TO Y_top STEP Y_step !                     ! \
1505   MOVE .8,I          ! Smidgeon left of Y-axis             !  > Label Y-axis
1510   LABEL USING "K";VAL$(I)&" "    !                         ! /
1515  NEXT I                          !                         !/
1520  PENUP
1525 ! Anisotropic scaling: left/right/bottom/top
1530  WINDOW X_left,X_right,Y_bottom,Y_top
1535  !
1540  ! --- get data to be plotted
1545  !
1550  READ Ndrifter
1551  ALLOCATE INTEGER Drifters(Ndrifter)
1552  READ Drifters(*)
1555  FOR Icurve=1 TO Ndrifter
1560   READ Input_file$
1565   Get_data(Input_file$,Id$,X_pos,Y_pos,Speed,Xoff,Yoff,Time,Ndata)
1570   IF Ndata THEN
1571    Id$=VAL$(Drifters(Icurve))
1575    Label(3.8,.6,0,4,1,X_pos,Y_pos+150,Id$)
1580    Label(3.8,.6,0,4,1,X_pos,Y_pos-150.,VAL$(INT(Speed*100)/100))
1585    Label(3,.6,0,4,1,X_pos,Y_pos,"+")
1590    PENUP
1595   END IF
1600  NEXT Icurve
1605  BEEP
1610  ON KBD GOTO Exit
1615  DISP "Enter 'Space bar' to go on"
1620 Idle:GOTO Idle                        ! View the plot as long as you want
1625 Exit:DISP
1630  IF Dump_graphics THEN
1635 ! DUMP GRAPHICS CRT TO #701
1640  END IF
1645  GRAPHICS OFF
1650  OUTPUT 2 USING "#,K";C$
1655  END
1660  !
1665 Gdu:SUB Gdu(X_gdu_max,Y_gdu_max,OPTIONAL Gdu_xmid,Gdu_ymid)
1670  ! This returns Xright, Yhigh and their respective midpoints in GDUs.
1675  ! Note that if Gud_xmid is defined, Gdu_ymid must be also.
1680   COM /G_units/ Gdu_xmax,Gdu_ymax,Udu_xmin,Udu_xmax,Udu_ymin,Udu_ymax,Show
1685   IF Gdu_xmax=0 THEN
1690    Gdu_xmax=100*MAX(1,RATIO)
```

```
1695    Gdu_ymax=100*MAX(1,1./RATIO)
1700   END IF
1705   X_gdu_max=Gdu_xmax
1710   Y_gdu_max=Gdu_ymax
1715   IF NPAR>2 THEN
1720    Gud_xmid=Gdu_xmax*.5
1725    Gud_ymid=Gdu_ymax*.5
1730   END IF
1735  SUBEND
1740 Label:SUB Label(Csize,Asp_ratio,Ldir,Lorg,Pen,X,Y,Text$)
1745  ! This defines several systems variables (in CSIZE, LDIR, etc.), and
1750  ! labels the test (if any) accordingly.
1755   DEG
1760   CSIZE Csize,Asp_ratio
1765   LDIR Ldir
1770   LORG Lorg
1775   PEN Pen
1780   MOVE X,Y
1785   IF Text$<>"" THEN LABEL USING "#,K";Text$
1790   PENUP
1795  SUBEND
1800  !
1805 Get_data:SUB Get_data(File_in$,Drno$,X_pos,Y_pos,Speed,Xoff,Yoff,Time,Ndata
)
1810  !
1815   OPTION BASE 1
1820   DIM Id$[11],Desc$[30]
1825   INTEGER Npos
1830   ASSIGN @Path_in TO File_in$
1835   ENTER @Path_in;Drno$,Id$,Desc$,Npos
1840   Ndata=0
1845   ON END @Path_in GOTO Close_file
1850   ENTER @Path_in;X_pos,Y_pos,T,Speed,Dummy        !   Change Dummy and Speed
 for U&V
1855   IF T<>Time THEN 1845
1860   X_pos=X_pos-Xoff
1865   Y_pos=Y_pos-Yoff
1870   Ndata=1
1875 Close_file:ASSIGN @Path_in TO *
1880  SUBEND
```

```
1000  ! PROGRAM: PLOT_PVS
1005  !
1010  ! --- Routine plots drifter's position and u-component speed.
1015  !
1020  ! ---------------- INPUT DATA SECTION ------------------------------
1025  ! Title$: Title of this plot
1030  DATA "Drifter Position & V_speed"
1035  ! ------------------------------------------------------------------
1040  ! X_label$: X_axis Label
1045  DATA "X-DIST. (M)"
1050  ! -------------------------------------------------------------------
1055  ! Y_label$: Y_axis label
1060  DATA "Y-DIST. (M)"
1065  ! -------------------------------------------------------------------
1070  !     X_left,X_right,X_tick_spacing,X_major_count
1075  DATA       0,    4000,            250,             2
1080  ! -------------------------------------------------------------------
1085  !     Y_bottom,  Y_top,Y_tick_spacing,Y_major_count
1090  DATA       0,    4000,            250,             2
1095  ! --------------------------------------------------------------------
1100  ! Grid: Need grid lines (yes/no; 1/0)
1105  DATA 0
1110  ! --------------------------------------------------------------------
1115  ! Dump_graphics: Dump graphics to printer (yes/no; 1/0)
1120  DATA 1
1125  ! --------------------------------------------------------------------
1130  ! Xoff,Yoff; offset in the x- and y- directions respectively.
1135  DATA 234500,349000
1140  ! --------------------------------------------------------------------
1145  ! Date_$: plotting date
1150  DATA 1 APR 1985
1155  ! --------------------------------------------------------------------
1160  ! Time_$: plotting time
1165  DATA 14:30:00
1170  ! --------------------------------------------------------------------
1175  ! Ndrifter: number of drifter to be plotted
1180  DATA 4
1185  ! --------------------------------------------------------------------
1190  ! Input_file$: input file name
1195  DATA "DRIFTER_5:,700,1"
1200  DATA "DRIFTER_7:,700,1"
1205  DATA "DRIFTER_8:,700,1"
1210  DATA "DRIFTER_9:,700,1"
1215  !
1220  ! --------------------- END OF INPUT DATA ------------------------
1225  !
1230  OPTION BASE 1
1235  DIM Title$[30],Id$[2]
1240  DIM Date_$[11],Time_$[8]
1245  READ Title$
```

```
1250  READ X_label$
1255  READ Y_label$
1260  READ X_left,X_right,X_tick_spacing,X_major_count
1265  READ Y_bottom,Y_top,Y_tick_spacing,Y_major_count
1270  READ Grid
1275  READ Dump_graphics
1280  READ Xoff,Yoff
1285  READ Date_$
1290  READ Time_$
1295  Time=DATE(Date_$)+TIME(Time_$)
1300  Major_tick_size=3.0
1305  C$=CHR$(255)&"K"
1310  OUTPUT 2 USING "#,K";C$        ! Clear screen for graph
1315  GINIT                          ! Initialize various graphics parameters.
1320  PLOTTER IS 3,"INTERNAL"        ! Use the internal screen
1325  GRAPHICS ON                    ! Turn on the graphics screen
1330  LORG 6                         ! Reference point: center of top of label
1335  ! Determine how many GDUs wide and high the screen is
1340  Gdu(X_gdu_max,Y_gdu_max)
1345  FOR I=-.2 TO .2 STEP .1        ! Offset of X from starting point
1350   MOVE X_gdu_max/2.3+I,Y_gdu_max ! Move to about middle of top of screen
1355   LABEL USING "#,K";Title$      ! Write title of plot
1360  NEXT I                         ! Next position for title
1365  DEG                            ! Angular mode is degrees (used in LDIR)
1370  Label(3.8,.6,90,6,1,0.,Y_gdu_max/2,Y_label$)   ! Write Y_axis label
1375  Label(3.8,.6,0,4,1,X_gdu_max/2,.01*Y_gdu_max,X_label$) ! Write X_axis labe
l
1380  Label(3.8,.6,0,4,1,.85*X_gdu_max,.8*Y_gdu_max,Date_$)
1385  Label(3.8,.6,0,4,1,.85*X_gdu_max,.75*Y_gdu_max,Time_$)
1390  Label(3.8,.6,0,4,1,.87*X_gdu_max,.7*Y_gdu_max,"X_off:"&VAL$(Xoff))
1395  Label(3.8,.6,0,4,1,.87*X_gdu_max,.65*Y_gdu_max,"Y_off:"&VAL$(Yoff))
1400  ! Define subset of screen area
1405  VIEWPORT .12*X_gdu_max,.75*X_gdu_max,.1*Y_gdu_max,.93*Y_gdu_max
1410 ! Anisotropic scaling: left/right/bottom/top
1415  WINDOW X_left,X_right,Y_bottom,Y_top
1420  ! Draw a box
1425  AXES X_tick_spacing,Y_tick_spacing,X_left,Y_bottom,X_major_count,Y_major_c
ount,Major_tick_size
1430  AXES X_tick_spacing,Y_tick_spacing,X_right,Y_top,X_major_count,Y_major_cou
nt,Major_tick_size
1435  IF Grid THEN GRID X_tick_spacing,Y_tick_spacing,X_left,Y_bottom,X_major_co
unt,Y_major_count
1440  CLIP OFF                       ! So labels can be outside VIEWPORT limits
1445  CSIZE 3.5,.6                   ! Smaller chars for axis labelling
1450  LORG 6                         ! Ref. pt: Top center      !\
1455  WINDOW X_left,X_right,.1*Y_gdu_max,.9*Y_gdu_max
1460  X_step=X_tick_spacing*X_major_count
1465  FOR I=X_left TO X_right STEP X_step! Every X_STEP units ! \
1470   MOVE I-X_left+1,.09*Y_gdu_max! A smidgeon below X-axis !  >  Label X-axis
1475   LABEL USING "#,K";I           !                          ! /
```

```
1480  NEXT I                          !                          !/
1485  WINDOW X_left,X_right,Y_bottom,Y_top
1490  LORG 8                          ! Ref. pt: Right center    !\
1495  Y_step=Y_tick_spacing*Y_major_count
1500  FOR I=Y_bottom TO Y_top STEP Y_step !                        ! \
1505   MOVE .8,I          ! Smidgeon left of Y-axis                !  > Label Y-axis
1510   LABEL USING "K";VAL$(I)&" "    !                            ! /
1515  NEXT I                          !                            !/
1520  PENUP
1525 ! Anisotropic scaling: left/right/bottom/top
1530  WINDOW X_left,X_right,Y_bottom,Y_top
1535  !
1540  ! --- get data to be plotted
1545  !
1550  READ Ndrifter
1555  FOR Icurve=1 TO Ndrifter
1560   READ Input_file$
1565   Get_data(Input_file$,Id$,X_pos,Y_pos,Speed,Xoff,Yoff,Time,Ndata)
1570   IF Ndata THEN
1575    Label(3.8,.6,0,4,1,X_pos,Y_pos+150,Id$)
1580    Label(3.8,.6,0,4,1,X_pos,Y_pos-150.,VAL$(INT(Speed*100)/100))
1585    Label(3,.6,0,4,1,X_pos,Y_pos,"+")
1590    PENUP
1595   END IF
1600  NEXT Icurve
1605  BEEP
1610  ON KBD GOTO Exit
1615  DISP "Enter 'Space bar' to go on"
1620 Idle:GOTO Idle                      ! View the plot as long as you want
1625 Exit:DISP
1630  IF Dump_graphics THEN
1635   DUMP GRAPHICS CRT TO #701
1640  END IF
1645  GRAPHICS OFF
1650  OUTPUT 2 USING "#,K";C$
1655  END
1660  !
1665 Gdu:SUB Gdu(X_gdu_max,Y_gdu_max,OPTIONAL Gdu_xmid,Gdu_ymid)
1670  ! This returns Xright, Yhigh and their respective midpoints in GDUs.
1675  ! Note that if Gud_xmid is defined, Gdu_ymid must be also.
1680   COM /G_units/ Gdu_xmax,Gdu_ymax,Udu_xmin,Udu_xmax,Udu_ymin,Udu_ymax,Show
1685   IF Gdu_xmax=0 THEN
1690    Gdu_xmax=100*MAX(1,RATIO)
1695    Gdu_ymax=100*MAX(1,1./RATIO)
1700   END IF
1705   X_gdu_max=Gdu_xmax
1710   Y_gdu_max=Gdu_ymax
1715   IF NPAR>2 THEN
1720    Gud_xmid=Gdu_xmax*.5
1725    Gud_ymid=Gdu_ymax*.5
```

```
1730   END IF
1735  SUBEND
1740 Label:SUB Label(Csize,Asp_ratio,Ldir,Lorg,Pen,X,Y,Text$)
1745  ! This defines several systems variables (in CSIZE, LDIR, etc.), and
1750  ! labels the test (if any) accordingly.
1755   DEG
1760   CSIZE Csize,Asp_ratio
1765   LDIR Ldir
1770   LORG Lorg
1775   PEN Pen
1780   MOVE X,Y
1785   IF Text$<>"" THEN LABEL USING "#,K";Text$
1790   PENUP
1795  SUBEND
1800  !
1805 Get_data:SUB Get_data(File_in$,Drno$,X_pos,Y_pos,Speed,Xoff,Yoff,Time,Ndata
)
1810  !
1815   OPTION BASE 1
1820   DIM Id$[1],Desc$[30]
1825   INTEGER Npos
1830   ASSIGN @Path_in TO File_in$
1835   ENTER @Path_in;Drno$,Id$,Desc$,Npos
1840   Ndata=0
1845   ON END @Path_in GOTO Close_file
1850   ENTER @Path_in;X_pos,Y_pos,T,Dummy,Speed
1855   IF T<>Time THEN 1845
1860   X_pos=X_pos-Xoff
1865   Y_pos=Y_pos-Yoff
1870   Ndata=1
1875 Close_file:ASSIGN @Path_in TO *
1880  SUBEND
```

```
1000  ! Program PREPARE
1005  DIM Id$[1],Desc$[30],Drid$[2]
1010  INTEGER Npos,I,Flag,Check,N
1015  REAL X,Y,T,T1,U,V
1020  DIM Length(14)
1025  DATA 28,45,64,38,51,99,100,78,139,159,164,186,165,47
1030  ASSIGN @Path_in TO "LDI01APR:,700,1"
1035  FOR Idr=1 TO 14
1040   Drid$=VAL$(Idr)
1045   READ Length(Idr)
1050   CREATE BDAT "DRIFTER_"&VAL$(Idr)&":,700,1",Length(Idr),80
1055   ASSIGN @Path_out TO "DRIFTER_"&VAL$(Idr)&":,700,1"
1060   ENTER @Path_in;Id$,Desc$,Npos
1065   PRINT Drid$,Id$,Desc$,Npos
1070   OUTPUT @Path_out;Idrid$,Id$,Desc$,Npos
1075   ENTER @Path_in;Flag,X1,Y1,T1
1080   ENTER @Path_in;Flag,X,Y,T
1085   U=(X-X1)/120.*100.
1090   V=(Y-Y1)/120.*100.
1095   X1=X
1100   Y1=Y
1105   T1=T
1110   PRINT X,T,U,V
1115   OUTPUT @Path_out;X,Y,T,U,V
1120   FOR I=3 TO Npos
1125    ENTER @Path_in;Flag,X,Y,T
1130    U=(X-X1)/(T-T1)*100.
1135    V=(Y-Y1)/(T-T1)*100.
1140    X1=X
1145    Y1=Y
1150    T1=T
1155    OUTPUT @Path_out;X,Y,T,U,V
1160    PRINT I,X,T,U
1165   NEXT I
1170 Next_idr:NEXT Idr
1175  END
```

```
1000  ! PROGRAM: PRINT_OA
1001  ! --- routne prints the Objective analysis results.
1005  OPTION BASE 1
1010  DIM Title1$[80],Title2$[80],A_$[256],App_$[256]
1015  DIM Phi(21,21),A(15,15)
1016  Clear$=CHR$(255)&"K"
1017  OUTPUT 2 USING "#,K";Clear$ ! clear the screen.
1020  ! ----------------------------------------------------------------
1025  ! assign the objective analysis results file to Oa_file$
1030  INPUT "Enter the filename of objective analysis file to be printed",Oa_fil
e$
1035  Oa_file$=UPC$(Oa_file$)
1040  ! ----------------------------------------------------------------
1045  ! read in the x- and y-grid points of the O. A. field
1050  INPUT "Enter the x- and y-grid points of the computing field(NPX,NPY)",Npx
,Npy
1055  IF Npx<=21 AND Npy<=21 THEN 1095
1060  BEEP
1065  PRINT TABXY(10,10),"                   ******* Warning *******"
1070  PRINT TABXY(10,11),"The sizes of array Phi(i,j) defined in this version"
1075  PRINT TABXY(10,12),"are incorrect, make correction in lines 1015 and 1055.
"
1080  PRINT TABXY(10,13),"Program execution is halted."
1085  STOP
1090  ! set the scaling factor for o.a. output
1095  INPUT "Enter the scaling factor for entire field data",Scaling_factor
1100  ! need a hard copy? (yes/no; 1/0)
1105  INPUT "Need a hard copy? (yes/no; 1/0)",Copy
1110  IF Copy THEN PRINTER IS 701
1115  ! ----------------------------------------------------------------
1120  ASSIGN @Path_in TO Oa_file$
1125  ON END @Path_in GOTO Close_file
1130  ENTER @Path_in;Title1$
1135  ENTER @Path_in;Title2$
1140  FOR I=Npy TO 1 STEP -1
1145   FOR J=1 TO Npx
1150    ENTER @Path_in;Phi(I,J)
1155   NEXT J
1160  NEXT I
1165  CALL Intrp(Phi(*),15,A(*))
1170  OUTPUT 2 USING "#,K";C$
1175  PRINT TAB(30),Title1$
1180  PRINT TAB(30),Title2$
1185  PRINT USING "K,SD.DE";"Scale:",Scaling_factor
1190  PRINT "       1     2     3     4     5     6     7     8     9    10    11    12    1
3   14    15"
1195  PRINT "       *     *     *     *     *     *     *     *     *     *     *     *
*     *     *"
1200  FOR I=1 TO 15
1205   A_$=VAL$(16-I)&"*"
```

```
1210   IF LEN(A_$)<3 THEN A_$=" "&A_$
1215   FOR J=1 TO 15
1220    App_$=VAL$(INT(A(I,J)/Scaling_factor))
1225    SELECT LEN(App_$)
1230    CASE 0
1235    CASE 1
1240     App_$="    "&App_$
1245    CASE 2
1250     App_$="   "&App_$
1255    CASE 3
1260     App_$="  "&App_$
1265    CASE 4
1270     App_$=" "&App_$
1275    END SELECT
1280    A_$=A_$&App_$
1285   NEXT J
1290   PRINT A_$
1295  NEXT I
1300  DISP "Enter 'CONTONUE' to go on"
1305  PAUSE
1310  DISP
1315  GOTO 1125
1320 Close_file:ASSIGN @Path_in TO *
1325  PRINTER IS CRT
1330  END
1335  !
1340  !
1345  !
1350 Intrp:SUB Intrp(B(*),M,A(*))
1355   OPTION BASE 1
1360   INTEGER Lli,Llip1,Llj,Lljp1
1365  ! Routnine interpolates array B(M,M) onto array A(15,15).
1370   FOR I=1 TO 15
1375    Lli=INT((M-1)*(I-1)/14.+.99999)
1380    Llip1=Lli+1
1385    IF Lli=0 THEN Lli=1
1390    Dil=(I-1)*(M-1)/14.-(Lli-1)
1395    Dir=1.-Dil
1400    FOR J=1 TO 15
1405     Llj=INT((M-1)*(J-1)/14.+.99999)
1410     Lljp1=Llj+1
1415     IF Llj=0 THEN Llj=1
1420     Djl=(J-1)*(M-1)/14.-(Llj-1)
1425     Dju=1.-Djl
1430     A(I,J)=(Dir*B(Lli,Llj)+Dil*B(Llip1,Llj))*Dju+(Dir*B(Lli,Lljp1)+Dil*B(Ll
ip1,Lljp1))*Djl
1435    NEXT J
1440   NEXT I
1445  SUBEND
```

```
1000  ! PROGRAM GET_IP
1005  ! ROUTINE CALCULATES INTERPOLATION POSITION
1010  !
1015  OPTION BASE 1
1020  RAD
1025  Xoff=234500
1030  Yoff=349000
1035  !Input dataport
1070  !
1075  ! --- create a BDAT file for output
1076  !        output_filename,file_length,record_size
1080  READ Filename$,File_length,Record_size
1081  DATA "IP_POS:,700,1",225,80
1082  ON ERROR GOTO 1084
1083  CREATE BDAT Filename$,File_length,Record_size
1084  OFF ERROR
1085  ASSIGN @Path_out TO Filename$
1090  !
1095  READ Npx,Npy   ! read in the number of grid points in x- and y-directions
1096  DATA 15,15
1100  FOR J=1 TO Npy
1105   Y=(J-1)*250.+Yoff
1110   FOR I=1 TO Npx
1115    X=(I-1)*250.+Xoff
1120    OUTPUT @Path_out;X,Y
1125    PRINT X,Y
1130   NEXT I
1135  NEXT J
1140  ASSIGN @Path_out TO *
1145  END
```

```
1000  ! PROGRAM SOA
1005  !
1010  ! ***********************************************************************
1015  ! *                                                                     *
1020  ! *              Scalar Space-Time Objective Analysis Package           *
1025  ! *                                                                     *
1030  ! * Language: BASIC 3.0                                                 *
1035  ! * System: Hewlett Packard 9816                                        *
1040  ! * Version: 1.00                                                       *
1045  ! * Date: July 1986                                                     *
1050  ! *                                                                     *
1055  ! * Revised by Dr. L. Charles Sun                                       *
1060  ! *              Hawaii Institute of Geophysics                         *
1065  ! *              University of Hawaii at Manoa                          *
1070  ! *              Honolulu, Hawaii 96822                                 *
1075  ! *                                                                     *
1080  ! * Developed at U. S. Coast Guard Research and Development Center      *
1085  ! *                 Groton, Connecticut 06340                           *
1090  ! *                                                                     *
1095  ! ***********************************************************************
1100  !
1105  !
1110  OPTION BASE 1
1115  RAD
1120  DIM Message$[256]
1125  DIM Date0$[15],Time0$[15]
1130  DIM Xdata(2000),Ydata(2000),Tdata(2000),Phidata(2000)
1135  DIM Xrd(1089),Yrd(1089),Trd(1089),Phird(1089)
1140  DIM Xopd(1089),Yopd(1089),Topd(1089),Phiopd(1089)
1145  DIM Xip(1089),Yip(1089),Tip(1089)
1150  DIM Theta(1089),Es(1089),Erms(1089)
1155  DIM Cor(1089),Index(1089)
1160  COM /Fold/ Xfold,Yfold,Tfold
1165  COM /Efield/ Evar
1170  COM /Cblock/ C(40)
1175  !
1180  Clear$=CHR$(255)&"K"
1185  Sdate$=DATE$(TIMEDATE)
1190  Stime$=TIME$(TIMEDATE)
1195  !
1200  OUTPUT KBD;Clear$;           ! clear the screen.
1205  !
1210  PRINT TABXY(15,10),"Do you need documentations?"
1215  INPUT "Enter Y/N for yes/no",Ans$
1220  OUTPUT KBD;Clear$;           ! clear the screen.
1225  IF UPC$(Ans$[1;1])<>"Y" THEN GOTO 1450
1230  Ty=10
1235  PRINT TABXY(10,7),"Which section do you want to look at ?"
1240  PRINT TAB(Ty),"Section 1: INTRODUCTION"
1245  PRINT TAB(Ty),"         2: BASIC THEORY"
```

```
1250  PRINT TAB(Ty),"         3: DETERMINATION OF CORRELATON FUNCTION"
1255  PRINT TAB(Ty),"         4: ELEMINATION OF DISTANT DATA"
1260  PRINT TAB(Ty),"         5: PROGRAMS DESCRIPTIONS"
1265  PRINT TAB(Ty),"         6: USER INSTRUCTIONS AND EXAMPLES"
1270  PRINT TAB(Ty),"         7: REFERENCES"
1275  PRINT TAB(Ty),"         8: DEFINITIONS OF VARIABLES"
1280  PRINT
1285  PRINT TAB(Ty),"Enter E to exit help manus, and start computation."
1290  INPUT "Selection?",Code$
1295  OUTPUT KBD;Clear$;
1300  IF Code$="E" OR Code$="e" THEN GOTO 1450
1305  File$="Doc_"&Code$
1310  ASSIGN @Read_message TO File$
1315 Read_message:  !
1320  Count=0
1325  ON END @Read_message GOTO End_mess
1330  ENTER @Read_message;Message$
1335  PRINT TAB(7),Message$[8;80]
1340  Count=Count+1
1345  IF Count=18 THEN
1350   INPUT "More message ? ",Ans$
1355   IF Ans$="y" OR Ans$="Y" THEN
1360    OUTPUT KBD;Clear$;
1365    GOTO Read_message
1370   ELSE
1375    OUTPUT KBD;Clear$;
1380    GOTO 1235
1385   END IF
1390  END IF
1395  GOTO 1325
1400 End_mess:  !
1405  INPUT "End of this section, Need another section? ",Ans$
1410  IF Ans$="y" OR Ans$="Y" THEN
1415   OUTPUT KBD;Clear$;
1420   GOTO 1235
1425  ELSE
1430   OUTPUT KBD;Clear$;
1435   GOTO 1450
1440  END IF
1445  !
1450 Input_parameter:  !
1455  OUTPUT KBD;Clear$;
1460  PRINT "*********************************************************************
***"
1465  PRINT "*
  *"
1470  PRINT "*         Scalar Space-Time Objective Anal,      ackage
  *"
1475  PRINT "*
  *"
```

```
1480  PRINT "*                        Language: BASIC 3.01
  *"
1485  PRINT "*                  System: Hewlett Packard 9000-2000
  *"
1490  PRINT "*                          Version 1.00
  *"
1495  PRINT "*                            July 1986
  *"
1500  PRINT "*
  *"
1505  PRINT "*              Coast Guard Research and Development Center
  *"
1510  PRINT "*                       Groton, Connecticut 06340
  *"
1515  PRINT "*
  *"
1520  PRINT "*******************************************************************
***"
1525  PRINT
1530  Tab=10
1535  PRINT "Before you go on, please check the followings:"
1540  PRINT "  1) Is the disc containing the observed data and"
1545  PRINT "     the inter/extrapolation positions in DRIVE# 1 ?"
1550  PRINT "  2) Are the output files existed ?, if not, create them before"
1555  PRINT "      you do the analysis."
1560  PRINT "  3) The file length for each output should be 2 records longer tha
n"
1565  PRINT "     the total inter/extrapolation position points."
1570  PRINT "  4) The current version allows 2000 input data points "
1575  PRINT "     and 1089 inter/extrapolation points."
1580  PRINT "  5) This version gets the correlation function from the"
1585  PRINT "     fitted formula."
1590 !PRINT
1595 !PRINT TAB(Tab),"Send your inquires to"
1600 !PRINT TAB(Tab),"     Dr. L. Charles Sun"
1605 !PRINT TAB(Tab),"     Deptartment of Oceanography"
1610 !PRINT TAB(Tab),"     University of Hawaii"
1615 !PRINT TAB(Tab),"     Honolulu, HI 96822"
1620 !PRINT TAB(Tab),"      (808)948-7633"
1625  INPUT "Hit [RETURN] or [ENTER] to continue",Answ$
1630  Tx=10
1635  Ty=10
1640  !
1645  Lp_flg=0
1650  LOOP
1655   OUTPUT KBD;Clear$;
1660  ! ----------------------------------------------------------------
1665  ! Date0$,Time0$ = time of the analysis to make.
1670   PRINT TABXY(Tx,Ty),"Enter time of the analysis to make.(DD MMM YY,HH:MM:S
S)"
```

```
1675   PRINT TAB(Ty),"(enclosed with quotations.)"
1680   DISP "Date0$?,Time0$?";
1685   IF Lp_flg THEN OUTPUT KBD;"""";Date0$;""","""";Time0$;"""";
1690   INPUT "",Date0$,Time0$
1695   Date0$=UPC$(Date0$)
1700   OUTPUT KBD;Clear$;
1705   ! ---------------------------------------------------------------------
1710   ! Nobj = number of objective analysis to make.
1715   PRINT TABXY(Tx,Ty),"Enter the number of objective analyses to make."
1720   DISP "Nobj?";
1725   IF Lp_flg THEN OUTPUT KBD;Nobj;
1730   INPUT "",Nobj
1735   OUTPUT KBD;Clear$;
1740   ! ---------------------------------------------------------------------
1745   ! Delta_t = time interval for each extrapolation in time.
1750   PRINT TABXY(Tx,Ty),"Enter the time interval for extrapolation in time"
1755   PRINT TAB(Ty),"   or zero for instantaneous computation."
1760   DISP "Delta_t?";
1765   IF Lp_flg THEN OUTPUT KBD;Delta_t;
1770   INPUT "",Delta_t
1775   OUTPUT KBD;Clear$;
1780   ! ---------------------------------------------------------------------
1785   ! Xlimit = Max. distance radius from the reference point of the domain.
1790   PRINT TABXY(Tx,Ty),"Enter maximum distance radius from the reference poin
t of the domain."
1795   PRINT TAB(Ty),"All data within this range are retained."
1800   DISP "Xlimit?";
1805   IF Lp_flg THEN OUTPUT KBD;Xlimit;
1810   INPUT "",Xlimit
1815   OUTPUT KBD;Clear$;
1820   ! ---------------------------------------------------------------------
1825   ! Tlimit = Max. time radius before and after the time of the analysis.
1830   PRINT TABXY(Tx,Ty),"Enter the max. time radius before and after the time
of the analysis."
1835   PRINT TAB(Ty),"All data within this range are retained."
1840   DISP "Tlimit?";
1845   IF Lp_flg THEN OUTPUT KBD;Tlimit;
1850   INPUT "",Tlimit
1855   OUTPUT KBD;Clear$;
1860   ! ---------------------------------------------------------------------
1865   ! Max_space_lag = maximum space lag.
1870   ! Max_time_lag = maximum time lag.
1875   PRINT TABXY(Tx,Ty),"Set the influence domain for computation points,"
1880   PRINT TAB(Ty),"The maximum time lags >= the multiplication of time interv
al"
1885   PRINT TAB(Ty),"      and the number of the analysis to make."
1890   PRINT TAB(Ty),"The maximum space lags >= the sizes of the computation dom
ain."
1395   PRINT
1900   PRINT TAB(Ty),"Enter the maximum space and time lags."
```

```
1905  DISP "Max_space_lag?,Max_time_lag?";
1910  IF Lp_flg THEN OUTPUT KBD;Max_space_lag,Max_time_lag;
1915  INPUT "",Max_space_lag,Max_time_lag
1920  OUTPUT KBD;Clear$;
1925  ! ------------------------------------------------------------------
1930  ! Limit = Max. number of influential points.
1935  PRINT TABXY(Tx,Ty),"Enter the maximum number of influential points (40 ma
x)."
1940  DISP "Limit?";
1945  IF Lp_flg THEN OUTPUT KBD;Limit;
1950  INPUT "",Limit
1955  OUTPUT KBD;Clear$;
1960  ! ------------------------------------------------------------------
1965  ! Xfold = X direction e-folding scale
1970  PRINT TABXY(Tx,Ty),"Enter the X direction E-folding scale"
1975  DISP "Xfold?";
1980  IF Lp_flg THEN OUTPUT KBD;Xfold;
1985  INPUT "",Xfold
1990  OUTPUT KBD;Clear$;
1995  !
2000  ! ------------------------------------------------------------------
2005  ! Yfold = Y direction e-folding scale
2010  PRINT TABXY(Tx,Ty),"Enter the Y direction E-folding scale."
2015  DISP "Yfold?";
2020  IF Lp_flg THEN OUTPUT KBD;Yfold;
2025  INPUT "",Yfold
2030  OUTPUT KBD;Clear$;
2035  !
2040  ! ------------------------------------------------------------------
2045  ! Tfold = Time e-folding scale
2050  PRINT TABXY(Tx,Ty),"Enter the Time E-folding scale (seconds)."
2055  DISP "Tfold?";
2060  IF Lp_flg THEN OUTPUT KBD;Tfold;
2065  INPUT "",Tfold
2070  OUTPUT KBD;Clear$;
2075  ! ------------------------------------------------------------------
2080  ! Data_file$ = Observed data file.
2085  PRINT TABXY(Tx,Ty),"Enter the observed data file including file specifier
s. "
2090  DISP "Data_file$?";
2095  IF Lp_flg THEN OUTPUT KBD;Data_file$;
2100  LINPUT "",Data_file$
2105  Data_file$=UPC$(TRIM$(Data_file$))
2110  OUTPUT KBD;Clear$;
2115  ! ------------------------------------------------------------------
2120  ! Ip_file$ = Interpolation position  data file.
2125  PRINT TABXY(Tx,Ty),"Enter the interpolation position file including file
specifiers."
2130  DISP "Ip_file$?";
2135  IF Lp_flg THEN OUTPUT KBD;Ip_file$;
```

```
2140   LINPUT "",Ip_file$
2145   Ip_file$=UPC$(TRIM$(Ip_file$))
2150   OUTPUT KBD;Clear$;
2155  ! ------------------------------------------------------------------
2160  ! Soa_fcst$ = SOA forecast field file.
2165   PRINT TABXY(Tx,Ty),"Enter the file specifier for OA forecast fields."
2170   DISP "Soa_fcst$?";
2175   IF Lp_flg THEN OUTPUT KBD;Soa_fcst$;
2180   LINPUT "",Soa_fcst$
2185   Soa_fcst$=UPC$(TRIM$(Soa_fcst$))
2190   OUTPUT KBD;Clear$;
2195  ! ------------------------------------------------------------------
2200  ! Soa_evar$ = SOA expected error file.
2205   PRINT TABXY(Tx,Ty),"Enter the file specifier for OA expected error fields
."
2210   DISP "Soa_evar$?";
2215   IF Lp_flg THEN OUTPUT KBD;Soa_evar$;
2220   LINPUT "",Soa_evar$
2225   Soa_evar$=UPC$(TRIM$(Soa_evar$))
2230   OUTPUT KBD;Clear$;
2235  ! ------------------------------------------------------------------
2240  !
2245  ! --- Echo check
2250  !
2255   Tab=10
2260   PRINT TABXY(10,2)
2265   PRINT TAB(Tab),"************   Echo check of input variables   **********
**"
2270   PRINT TAB(Tab),"Time of the analysis:                         ";Date0$;" ";Time0
$
2275   PRINT TAB(Tab),"Number of objective analysis to make: ";Nobj
2280   PRINT TAB(Tab),"Time interval:                         ";Delta_t
2285   PRINT TAB(Tab),"Max. distance radius:                  ";Xlimit
2290   PRINT TAB(Tab),"Max. time radius:                      ";Tlimit
2295   PRINT TAB(Tab),"Maximum time lag:                      ";Max_time_lag
2300   PRINT TAB(Tab),"Maximum space lag:                     ";Max_space_lag
2305   PRINT TAB(Tab),"Maximum number of influential points: ";Limit
2310   PRINT TAB(Tab),"X direction E-folding scale            ";Xfold
2315   PRINT TAB(Tab),"Y direction E-folding scale            ";Yfold
2320   PRINT TAB(Tab),"Time E-folding scale                   ";Tfold
2325   PRINT TAB(Tab),"The observed data file:                 ";Data_file$
2330   PRINT TAB(Tab),"The interpolated positions file:        ";Ip_file$
2335   PRINT TAB(Tab),"The SOA forecast output file:           ";Soa_fcst$
2340   PRINT TAB(Tab),"The SOA expected error output file:     ";Soa_evar$
2345   Lp_flg=1
2350   Answ$=""
2355   INPUT "Do you want to change anv values (Y/N)  <no>",Answ$
2360  EXIT IF UPC$(Answ$[1;1])<>"Y"
2365  END LOOP
2370  !
```

```
2375  OUTPUT KBD;Clear$;
2380  !
2385  ! read in the observed data
2390  !
2395  Get_data(Data_file$,Xdata(*),Ydata(*),Tdata(*),Phidata(*),Ndata)
2400  IF Ndata=0 THEN
2405   PRINT TAB(Tab),"Error in reading the observed data; Program stopped"
2410   STOP
2415  END IF
2420  PRINT TAB(Tab),"Total number of the observed data: ",Ndata
2425  !
2430  ! read in the interpolated positions
2435  !
2440  Get_ip(Ip_file$,Xip(*),Yip(*),Nip)
2445  !
2450  IF Nip=0 THEN
2455   PRINT TAB(Tab),"Error in reading interpolated positions; Program stopped"
2460   STOP
2465  END IF
2470  PRINT "Total number of the interpolated positions:",Nip
2475  PRINT "Open ",Soa_fcst$," for forecast output."
2480  ASSIGN @Fcst_out TO Soa_fcst$
2485  PRINT "Open ",Soa_evar$," for expected error output."
2490  ASSIGN @Evar_out TO Soa_evar$
2495  WAIT 1
2500  !
2505  T0=DATE(Date0$)+TIME(Time0$) ! Convert the real time to HP time format
2510  !
2515  ! Loop for each analysis
2520  !
2525  FOR Iobj=1 TO Nobj
2530   OUTPUT KBD;Clear$;
2535   Tf=T0+Delta_t*Iobj
2540   Tl=Tf-Tlimit
2545   Tu=Tf+Tlimit
2550   IF Delta_t=0 THEN
2555    PRINT TABXY(Tx,Ty),"Time of the analysis to make: ",DATE$(Tf),TIME$(Tf)
2560   ELSE
2565    PRINT TABXY(Tx,Ty),"Forecast time: ",DATE$(Tf),TIME$(Tf)
2570   END IF
2575  !
2580  ! get the observed data points within the limited range and at the proper
time
2585  !
2590   Get_rd(Xdata(*),Ydata(*),Tdata(*),Phidata(*),Ndata,Tl,Tu,Xlimit,Xrd(*),Yr
d(*),Trd(*),Phird(*),N)
2595  !
2600   Evar=0.
2605   IF Evar>1.0 THEN
2610    PRINT TAB(Tab),"The error noise level 100% exceeded"
```

```
2615     STOP
2620    END IF
2625    !
2630    ! Do the analysis for each point
2635    !
2640    FOR Ip=1 TO Nip
2645     CALL Select(Limit,Xip(Ip),Yip(Ip),Tf,Xrd(*),Yrd(*),Trd(*),Phird(*),Xopd(
*),Yopd(*),Topd(*),Phiopd(*),N,Nobs,Max_time_lag,Max_space_lag)
2650     IF Nobs<>0 THEN 2670
2655     Theta(Ip)=99999
2660     Es(Ip)=99999
2665     GOTO Next_ip
2670     CALL Scalar_oa(Xopd(*),Yopd(*),Topd(*),Phiopd(*),Nobs,X,Y,Tf,Valp,W,Ier)
2675    !
2680    ! if the error field is too large, increment error variance and re_do
2685    !
2690     IF Ier=0 THEN GOTO 2705
2695     Evar=Evar+.01
2700     GOTO 2605
2705     Theta(Ip)=Valp
2710     Es(Ip)=SQR(ABS(W))
2715     DISP USING "K,ZZZZ,K,MD.DDDE,K,MD.DDDE,";"Point no. ",Ip," Theta=",Theta
(Ip)," Expected Errors=",Es(Ip)
2720 Next_ip:NEXT Ip
2725    !
2730    Tab=10
2735    PRINT TAB(Tab)
2740    PRINT TAB(Tab),"The diagnostics of the observed fields"
2745    CALL Diag(Phird(*),N)
2750    PRINT TAB(Tab),
2755    PRINT TAB(Tab),"the diagnostics of the predicted fields"
2760    CALL Diag(Theta(*),Nip)
2765    PRINT TAB(Tab),
2770    PRINT TAB(Tab),"the diagnostics of the error fields"
2775    CALL Diag(Es(*),Nip)
2780    !
2785    ! output the O.A .field
2790    !
2795    BEEP
2800     !
2805     ! --- output the O. A. forcast field.
2810     !
2815    DISP "Output the Forecast field  to ",Soa_fcst$
2820    OUTPUT @Fcst_out;"O. A. Forecast Field"
2825    OUTPUT @Fcst_out;"Time: "&DATE$(Tf)&"  "&TIME$(Tf)
2830    FOR Ip=1 TO Nip
2835     OUTPUT @Fcst_out;Theta(Ip)
2840    NEXT Ip
2845     !
2850     ! --- output the Expect Error field.
```

```
2855   !
2860   DISP "Output the error field  to ",Soa_evar$
2865   OUTPUT @Evar_out;"Expected Error Field"
2870   OUTPUT @Evar_out;"Time: "&DATE$(Tf)&"  "&TIME$(Tf)
2875   FOR Ip=1 TO Nip
2880    OUTPUT @Evar_out;Es(Ip)
2885   NEXT Ip
2890   DISP
2895   !
2900  NEXT Iobj
2905  !
2910  ! close all output files.
2915  !
2920  ASSIGN @Fcst_out TO *
2925  ASSIGN @Evar_out TO *
2930 Finish:  !
2935  DISP "Finished"
2940  END
2945  !
2950  !
2955  !
2960 Get_rd:SUB Get_rd(Xdata(*),Ydata(*),Tdata(*),Phidata(*),M,Tl,Tu,Xlimit,Xrd(
*),Yrd(*),Trd(*),Phird(*),N)
2965  !
2970  ! --- routine gets the data before and after a given time and within
2975  !      a given spatial radius from the domain reference point.
2980  !
2985   RAD
2990   OPTION BASE 1
2995   IF Tl=Tu THEN
3000    PRINT USING "K,K,K";"Use data on ",DATE$(Tu)," ",TIME$(Tu)
3005   ELSE
3010    PRINT USING "K,K,K,K,K,K,K,K";"Use data after ",DATE$(Tl)," ",TIME$(Tl),
" and before ",DATE$(Tu)," ",TIME$(Tu)
3015   END IF
3020   PRINT USING "K,MD.DDDE,K";" and within the range of ",Xlimit," from the r
eference point."
3025   N=0
3030   FOR I=1 TO M
3035    IF Tdata(I)>Tu THEN GOTO Next_i
3040    IF Tdata(I)<Tl THEN GOTO Next_i
3045    IF ABS(Xdata(I))>Xlimit THEN GOTO Next_i
3050    IF ABS(Ydata(I))>Xlimit THEN GOTO Next_i
3055    N=N+1
3060    Xrd(N)=Xdata(I)
3065    Yrd(N)=Ydata(I)
3070    Trd(N)=Tdata(I)
3075    Phird(N)=Phidata(I)
3080 Next_i:NEXT I
3085   IF N=0 THEN
```

```
3090    PRINT "No data found; Please re-define time or distance radius; Program
stopped."
3095     BEEP
3100     STOP
3105    ELSE
3110     PRINT "Number of data within this range:",N
3115    END IF
3120   SUBEND
3125   !
3130   !
3135   !
3140 Select:SUB Select(Limit,X,Y,Tf,Xrd(*),Yrd(*),Trd(*),Phird(*),Xopd(*),Yopd(*
),Topd(*),Phiopd(*),N,Nobs,Max_time_lag,Max_space_lag)
3145   !
3150   ! --- routne eliminates the distant (in space and time) points and selects
3155   !      the most "LIMIT" near points to an interpolation point X,Y,Tf.
3160   !
3165    OPTION BASE 1
3170    RAD
3175    DIM Index(1089),Cor(1089)
3180    COM /Fold/ Xfold,Yfold,Tfold
3185    COM /Efield/ Evar
3190    COM /Cblock/ C(40)
3195    Cphase=0.
3200    Nobs=0
3205    FOR I=1 TO N
3210     Delt=Tf-Trd(I)
3215     IF ABS(Delt)>Max_time_lag THEN GOTO Next_i
3220     Delx=X-Xrd(I)
3225     Dely=Y-Yrd(I)
3230     R=SQR((Delx-Cphase*Delt)*(Delx-Cphase*Delt)+Dely*Dely)
3235     IF R>Max_space_lag THEN GOTO Next_i
3240     Nobs=Nobs+1
3245     Index(Nobs)=I
3250     Cor(Nobs)=FNCov(Delx,Dely,Delt,Xfold,Yfold,Tfold)
3255 Next_i:NEXT I
3260    IF Nobs=0 THEN
3265     BEEP
3270     OUTPUT KBD;CHR$(255);"K";
3275     PRINT TABXY(10,10),"*** Warning.  .arning! Warning! ***"
3280     PRINT USING "K,K,K";"No data were selected for interpolated position: ",
X,Y
3285     PRINT "The present radii of the influence domain are:"
3290     PRINT "Maximum space lags: ",Max_space_lag
3295     PRINT "maximum time lags:  ",Max_time_lag
3300     PRINT "User responses:"
3305     PRINT "    1) Use larger maximum time or space lags, then start over."
3310     PRINT "    2) Assign a value of 99999 to the estimated value at this posi
tion and go to the next point."
3315     INPUT "Enter your option (1 or 2)  <2>",Option
```

```
3320     OUTPUT KBD;CHR$(255);"K";
3325     IF Option=1 THEN STOP
3330     SUBEXIT
3335    END IF
3340    IF Nobs>Limit THEN
3345 !   REDIM Cor(Nobs),Index(Nobs)
3350 !   MAT SORT Cor(*) DES TO Index
3355     Sort(Cor(*),Index(*),Nobs)
3360     Nobs=Limit
3365    END IF
3370    FOR I=1 TO Nobs
3375     J=Index(I)
3380     Xopd(I)=Xrd(J)
3385     Yopd(I)=Yrd(J)
3390     Topd(I)=Trd(J)
3395     Phiopd(I)=Phird(J)
3400     C(I)=Cor(I)
3405    NEXT I
3410   SUBEND
3415   !
3420   !
3425   !
3430 Scalar_oa:SUB Scalar_oa(Xopd(*),Yopd(*),Topd(*),Phiopd(*),N,X,Y,Tf,Valp,W,I
er)
3435   !
3440   ! --- The scalar space-time objective analysis routine.
3445   !
3450    OPTION BASE 1
3455    RAD
3460    DIM A(40,40)
3465    COM /Cblock/ C(40)
3470    COM /Efield/ Evar
3475    IF N<=0 THEN GOTO 3625
3480    CALL Set_inva(A(*),Xopd(*),Yopd(*),Topd(*),N,Ier)
3485    IF Ier>0 THEN SUBEXIT
3490    CALL Est_mean(A(*),Phiopd(*),N,Ave,Sum2)
3495    !
3500    ! W is the error variance
3505    !
3510    W=0.
3515    W2=0.
3520    !
3525    ! --- valp is the inpterpolated data
3530    !
3535    Valp=0.
3540    FOR I=1 TO N
3545     H=0.
3550     Dumc=C(I)
3555     FOR J=1 TO N
3560      P=Dumc*C(J)*A(I,J)
```

```
3565      W=W+P
3570      P2=A(I,J)*Phiopd(J)
3575      H=H+P2
3580      P3=C(J)*A(J,I)
3585      W2=W2+P3
3590     NEXT J
3595     Dumy=Dumc*H
3600     Valp=Valp+Dumy
3605    NEXT I
3610    Valp=Valp+Ave
3615    Wm=(1.-W2)^2/Sum2
3620    W=(1.-W)+Wm
3625   SUBEND
3630   !
3635   !
3640   !
3645 Set_inva:SUB Set_inva(A(*),Xopd(*),Yopd(*),Topd(*),Nobs,Ier)
3650 !
3655 ! --- routine sets up the correlation function for the observations
3660 !     given the positions Xopd, Yopd and times, topd, it returns the
3665 !     inverted correlation function matrix.
3670 !
3675    OPTION BASE 1
3680    RAD
3685    DIM Ip(1089)
3690    COM /Fold/ Xfold,Yfold,Tfold
3695    COM /Efield/ Evar
3700    Guard=1.0E-30
3705    Ier=0
3710    FOR I=1 TO Nobs
3715     FOR J=I TO Nobs
3720      Delt=Topd(I)-Topd(J)
3725      Delx=Xopd(I)-Xopd(J)
3730      Dely=Yopd(I)-Yopd(J)
3735      A(I,J)=FNCov(Delx,Dely,Delt,Xfold,Yfold,Tfold)
3740      A(J,I)=A(I,J)
3745     NEXT J
3750     A(I,I)=A(I,I)+Evar
3755    NEXT I
3760   !
3765   ! invert the matrix a
3770   !
3775    CALL Invmtx(A(*),Nobs,A(*),Nobs,Nobs,D,Ip(*),Ier)
3780    IF D<Guard THEN
3785     PRINT "*** WARNING THE DETERMINANT IS VERY SMALL; DET=",D
3790     PRINT "Suggestion to  user:  Use smaller max. influential points."
3795     Ier=-1
3800    END IF
3805   SUBEND
3810   !
```

```
3815  !
3820  !
3825 Sort:SUB Sort(Cor(*),Index(*),N)
3830  !
3835  ! --- routine sorts the index and correlation in descending order.
3840  !
3845   OPTION BASE 1
3850   RAD
3855   Igap=N
3860   IF Igap<=1 THEN SUBEXIT
3865   Igap=Igap/2
3870   Imax=N-Igap
3875   Iex=0
3880   FOR I=1 TO Imax
3885    Iplusg=I+Igap
3890    IF Cor(I)>=Cor(Iplusg) THEN GOTO Next_i
3895    Save=Cor(I)
3900    Cor(I)=Cor(Iplusg)
3905    Cor(Iplusg)=Save
3910    Isave=Index(I)
3915    Index(I)=Index(Iplusg)
3920    Index(Iplusg)=Isave
3925    Iex=1
3930 Next_i:NEXT I
3935   IF Iex<>0 THEN GOTO 3875
3940   GOTO 3860
3945  SUBEND
3950  !
3955  !
3960  !
3965 Est_mean:SUB Est_mean(A(*),Psi(*),N,Ave,Sum2)
3970  !
3975  ! --- routine calculates the estimated mean, and then removes
3980  !      the estimated mean from the observations array.
3985  !
3990   OPTION BASE 1
3995   RAD
4000   DIM C(1089),D(1089)
4005   FOR I=1 TO N
4010    C(I)=0.
4015    D(I)=0.
4020    FOR K=1 TO N
4025     C(I)=C(I)+A(I,K)*Psi(K)
4030     D(I)=D(I)+A(I,K)
4035    NEXT K
4040   NEXT I
4045   Sum1=0.
4050   Sum2=0.
4055   FOR I=1 TO N
4060    Sum1=Sum1+C(I)
```

```
4065    Sum2=Sum2+D(I)
4070   NEXT I
4075   !
4080   ! --- calculate the estimated mean
4085   !
4090   Ave=Sum1/Sum2
4095   !
4100   ! --- remove the estimated mean from the observations.
4105   !
4110   FOR I=1 TO N
4115    Psi(I)=Psi(I)-Ave
4120   NEXT I
4125  SUBEND
4130  !
4135  !
4140  !
4145 Invmtx:SUB Invmtx(A(*),Na,V(*),Nv,N,D,Ip(*),Ier)
4150  !
4155  ! --- routne inverts the Matrix A.
4160  !
4165  !     V is the inverted matrix of A.
4170  !     D is the determinent of Matrix a.
4175  !
4180   OPTION BASE 1
4185   RAD
4190   Iexmax=75
4195   Ier=FNIerinv(N,Na,Nv)
4200   IF Ier<>0 THEN 4555
4205   FOR J=1 TO N
4210    Ip(J)=0
4215    FOR I=1 TO N
4220     V(I,J)=A(I,J)
4225    NEXT I
4230   NEXT J
4235   D=1.
4240   Iex=0
4245   FOR M=1 TO N
4250    Vmax=0.
4255    FOR J=1 TO N
4260     IF Ip(J)<>0 THEN 4305
4265     FOR I=1 TO N
4270      IF Ip(I)<>0 THEN 4300
4275      Vh=ABS(V(I,J))
4280      IF Vmax>Vh THEN 4300
4285      Vmax=Vh
4290      K=I
4295      L=J
4300     NEXT I
4305    NEXT J
4310    Ip(L)=K
```

```
4315    Npm=N+M
4320    Ip(Npm)=L
4325    D=D*V(K,L)
4330    IF ABS(D)<=1.0 THEN 4350
4335    D=D*.1
4340    Iex=Iex+1
4345    GOTO 4330
4350    Pvt=V(K,L)
4355    IF M=1 THEN Pvtmx=ABS(Pvt)
4360    IF (ABS(Pvt/M)+Pvtmx)=Pvtmx THEN 4520
4365    V(K,L)=1.
4370    FOR J=1 TO N
4375     Hold=V(K,J)
4380     V(K,J)=V(L,J)
4385     V(L,J)=Hold/Pvt
4390    NEXT J
4395    FOR I=1 TO N
4400     IF I=L THEN 4430
4405     Hold=V(I,L)
4410     V(I,L)=0.
4415     FOR J=1 TO N
4420      V(I,J)=V(I,J)-V(L,J)*Hold
4425     NEXT J
4430    NEXT I
4435   NEXT M
4440   M=N+N+1
4445   FOR J=1 TO N
4450    M=M-1
4455    L=Ip(M)
4460    K=Ip(L)
4465    IF K=L THEN 4500
4470    D=-D
4475    FOR I=1 TO N
4480     Hold=V(I,L)
4485     V(I,L)=V(I,K)
4490     V(I,K)=Hold
4495    NEXT I
4500   NEXT J
4505   IF Iex>Iexmax THEN 4540
4510   D=D*10.^Iex
4515   GOTO 4555
4520   Ier=33
4525   BEEP
4530   PRINT TAB(15),"WARNING: MATRIX SINGULAR IN INVMTX"
4535   GOTO 4555
4540   Ier=1
4545   D=Iex
4550   PRINT TAB(15),"WARNING: DETERMINANT TOO LARGE IN INVMTX"
4555  SUBEND
4560  !
```

```
4565 Fnierinv:DEF FNIerinv(N,Na,Nv)
4570   OPTION BASE 1
4575   RAD
4580   Ierinv=0
4585   IF N>=1 THEN 4605
4590   Ierinv=34
4595   PRINT "N < 1 IN INVMTX"
4600   GOTO 4640
4605   IF Na>=N THEN 4625
4610   Ierinv=35
4615   PRINT "NA<N IN INVMTX"
4620   GOTO 4640
4625   IF Nv>=N THEN 4640
4630   Ierinv=36
4635   PRINT "NV < N IN INVMTX"
4640   RETURN Ierinv
4645  FNEND
4650  !
4655  !
4660  !
4665 Diag:SUB Diag(Dpsi(*),M)
4670  !
4675  ! --- routine calculates the statistical parameters of input array
4680  !     such as the mean, variance, root mean square, mimimum and
4685  !     maximum.
4690  !
4695   OPTION BASE 1
4700   RAD
4705  !
4710   IF M<=0 THEN SUBEXIT
4715   PRINT "Number of points: ",M
4720   Ave=0.
4725   Sdv=0.
4730   Psimax=Dpsi(1)
4735   Psimin=Dpsi(1)
4740   Ave=Dpsi(1)
4745   IF M=1 THEN Cal_rms
4750   FOR I=2 TO M
4755    IF Psimax<Dpsi(I) THEN Psimax=Dpsi(I)
4760    IF Psimin>Dpsi(I) THEN Psimin=Dpsi(I)
4765    Sdv=((I-2)*Sdv+(I-1)*(Dpsi(I)-Aver)^2/I)/(I-1)
4770    Ave=((I-1)*Ave+Dpsi(I))/I
4775   NEXT I
4780 Cal_rms:Rms=SQR(Sdv+Ave^2)
4785   PRINT "Mean and Variance: ",Ave,Sdv
4790   Sdv=SQR(Sdv)
4795   PRINT "Standard Deviation: ",Sdv
4800   PRINT "RMS of field: ",Rms
4805   PRINT "Minimum of field: ",Psimin
4810   PRINT "Maximum of field: ",Psimax
```

```
4815  SUBEND
4820  !
4825  !
4830  !
4835 Get_data:SUB Get_data(Filename$,Xdata(*),Ydata(*),Tdata(*),Phidata(*),Ndata
)
4840  !
4845  ! --- routine reads in the observed data.
4850  !
4855   ASSIGN @Path_in TO Filename$
4860   Ndata=0
4865   PRINT "       **********   Echo check of the first ten records    **********
*"
4870   PRINT "  No.      X_POS            Y_POS                TIME             DATA"
4875   ON END @Path_in GOTO Close_file
4880   DISP USING "K,K,K";"Reading the observed data from ",Filename$,", please
wait."
4885   LOOP
4890    ENTER @Path_in;X,Y,T,Phi,Dummy
4895    IF Ndata<=10 THEN PRINT USING "DDDD,XX,MD.DDDDDDE,XX,MD.DDDDDDE,XX,MD.DD
DDDDDDE,XX,MD.DDDDDDE";Ndata,X,Y,T,Phi
4900    Ndata=Ndata+1
4905    Xdata(Ndata)=X
4910    Ydata(Ndata)=Y
4915    Tdata(Ndata)=T
4920    Phidata(Ndata)=Phi
4925   END LOOP
4930 Close_file:ASSIGN @Path_in TO *  ! Closing input_file.
4935   DISP
4940  SUBEND
4945  !
4950  !
4955  !
4960 Get_ip:SUB Get_ip(Filename$,Xdata(*),Ydata(*),Nip)
4965  !
4970  ! --- routine reads in the computating points.
4975  !
4980   ASSIGN @Path_in TO Filename$
4985   PRINT "* Echo check of the first ten records *"
4990   PRINT "  No.   X_IP_POS.      Y_IP_POS."
4995   Nip=0
5000   ON END @Path_in GOTO Close_file
5005   DISP USING "K,K,K";"Reading the interpolated positions from ",Filename$,"
, please wait."
5010   LOOP
5015    ENTER @Path_in;X,Y
5020    IF Nip<=10 THEN PRINT USING "DDDD,XX,MD.DDDDDDE,XX,MD.DDDDDDE";Nip,X,Y
5025    Nip=Nip+1
5030    Xdata(Nip)=X
5035    Ydata(Nip)=Y
```

```
5040   END LOOP
5045 Close_file:ASSIGN @Path_in TO *   ! Closing input_file.
5050   DISP
5055  SUBEND
5060  DEF FNCov(Delx,Dely,Delt,Xfold,Yfold,Tfold)
5065  !
5070  ! routine gets the correlation function from a fitted formula.
5075  !
5080   OPTION BASE 1
5085   RAD
5090   COM /Efield/ Evar
5095   RETURN EXP(-SQR((Delx/Xfold)*(Delx/Xfold)+(Dely/Yfold)*(Dely/Yfold)+(Delt
/Tfold)*(Delt/Tfold)))
5100 ! R2=SQR((Delx^2+Dely^2)/5.0E+4^2)
5105 ! Cor=EXP(-R2)
5110 ! RETURN Cor
5115  FNEND
```

```
1000  ! PROGRAM VOA
1001  !  CHANGES TO PROGRAM BY A.ALLEN AND M.D.COUTURIER 16 DEC 1987
1005  !
1010  ! *****************************************************************
1015  ! *                                                               *
1020  ! *                VECTOR SPACE-TIME OBJECTIVE ANALYSIS           *
1025  ! *                                                               *
1030  ! * Language: BASIC 3.0                                           *
1035  ! * System: Hewlett Packard 9816                                  *
1040  ! * Version: 1.00                                                 *
1045  ! * Date: July 1986                                               *
1050  ! *                                                               *
1055  ! * Revisied by Dr. L. Charles Sun                                *
1060  ! *             Hawaii Institute of Geophysics                    *
1065  ! *             University of Hawaii                              *
1070  ! *             Honolulu, Hawaii 96822                            *
1075  ! *                                                               *
1080  ! * Developed at U. S. Coast Guard Research And Development Center *
1085  ! *              Grotin, Connecticut 06340                        *
1090  ! *                                                               *
1095  ! *****************************************************************
1100  !
1105  OPTION BASE 1
1110  RAD
1115  DIM Message$[256]
1120  DIM Xdata(1089),Ydata(1089),Tdata(1089),Udata(1089),Vdata(1089)
1125  DIM Xrd(1089),Yrd(1089),Trd(1089),Urd(1089),Vrd(1089)
1130  DIM Xopd(1089),Yopd(1089),Topd(1089),Uopd(1089),Vopd(1089)
1135  DIM Xip(1089),Yip(1089),Tip(1089)
1140  DIM Uest(1089),Vest(1089),Uerror(1089),Verror(1089),Unrms(1089),Vnrms(1089
)
1145  COM /Corr/ Cuu(33,33),Cuv(33,33),Cvv(33,33),Xbin,Ybin,Mbin,Nbin
1146  COM /Fold/ Xfold,Yfold,Tfold
1150  COM /Efield/ E1,E2
1155  Clear$=CHR$(255)&"K"
1160  Sdate$=DATE$(TIMEDATE)
1165  Stime$=TIME$(TIMEDATE)
1170  !
1175  OUTPUT KBD;Clear$;        ! clear the screen.
1180  !
1185  PRINT TABXY(15,10),"Do you need documentations?"
1190  INPUT "Enter Y/N for yes/no  <no>",Ans$
1195  OUTPUT KBD;Clear$;         ! clear the screen.
1200  IF UPC$(Ans$[1;1])<>"Y" THEN GOTO 1425
1205  Ty=10
1210  PRINT TABXY(10,7),"Which section do you want to look at ?"
1215  PRINT TAB(Ty),"Section 1: INTRODUCTION"
1220  PRINT TAB(Ty),"         2: BASIC THEORY"
1225  PRINT TAB(Ty),"         3: DETERMINATION OF CORRELATON FUNCTION"
1230  PRINT TAB(Ty),"         4: ELEMINATION OF DISTANT DATA"
```

```
1235  PRINT TAB(Ty),"         5: PROGRAMS DESCRIPTIONS"
1240  PRINT TAB(Ty),"         6: USER INSTRUCTIONS AND EXAMPLES"
1245  PRINT TAB(Ty),"         7: REFERENCES"
1250  PRINT TAB(Ty),"         8: DEFINITIONS OF VARIALBES"
1255  PRINT
1260  PRINT TAB(Ty),"Enter E to exit help manus, and start computation."
1265  INPUT "Selection?",Code$
1270  OUTPUT KBD;Clear$;
1275  IF Code$="E" OR Code$="e" THEN GOTO 1425
1280  File$="Doc_"&Code$
1285  ASSIGN @Read_message TO File$
1290 Read_message:  !
1295  Count=0
1300  ON END @Read_message GOTO End_mess
1305  ENTER @Read_message;Message$
1310  PRINT TAB(7),Message$[8;80]
1315  Count=Count+1
1320  IF Count=18 THEN
1325   INPUT "More message ? ",Ans$
1330   IF Ans$="y" OR Ans$="Y" THEN
1335    OUTPUT KBD;Clear$;
1340    GOTO Read_message
1345   ELSE
1350    OUTPUT KBD;Clear$;
1355    GOTO 1210
1360   END IF
1365  END IF
1370  GOTO 1300
1375 End_mess:  !
1380  INPUT "End of this section, Need another section? ",Ans$
1385  IF Ans$="y" OR Ans$="Y" THEN
1390   OUTPUT KBD;Clear$;
1395   GOTO 1210
1400  ELSE
1405   OUTPUT KBD;Clear$;
1410   GOTO 1425
1415  END IF
1420  !
1425 Input_parameter:  !
1430  Tab=7
1435  PRINT TAB(Tab)
1440  PRINT TAB(Tab)
1445  PRINT TAB(Tab)
1450  PRINT TAB(Tab),"*************************************************************
************"
1455  PRINT TAB(Tab),"*
          *"
1460  PRINT TAB(Tab),"*          Vector Space-Time Objective Analysis Package
          *"
1465  PRINT TAB(Tab),"*
```

```
        *"
1470  PRINT TAB(Tab),"*                          Language: BASIC 3.01
        *"
1475  PRINT TAB(Tab),"*                  System: Hewlett Packard 9000-2000
        *"
1480  PRINT TAB(Tab),"*                            Version 1.00
        *"
1485  PRINT TAB(Tab),"*                              July 1986
        *"
1490  PRINT TAB(Tab),"*
        *"
1495  PRINT TAB(Tab),"*                  Coast Guard Research and Development Center
        *"
1500  PRINT TAB(Tab),"*                  Groton, Connecticut 06340
        *"
1505  PRINT TAB(Tab),"*
        *"
1510  PRINT TAB(Tab),"*****************************************************
************"
1515  WAIT 1
1520  OUTPUT KBD;Clear$;
1525  PRINT
1530  PRINT
1535  Tab=10
1540  PRINT TAB(Tab),"Before you go on, please check the followings:"
1545  PRINT TAB(Tab),"  1) Is the disc containing the observed data and"
1550  PRINT TAB(Tab),"     the inter/extraplotation positions in DRIVE# 1 ?"
1555  PRINT TAB(Tab),"  2) are the output files existed ?, if not, create them b
efore"
1560  PRINT TAB(Tab),"      you do the analysis."
1565  PRINT TAB(Tab),"  3) The file length for each output should be 2 records l
onger than"
1570  PRINT TAB(Tab),"     the total inter/extrapolation position points."
1575  PRINT TAB(Tab),"  4) The current version allows 2000 input data and"
1580  PRINT TAB(Tab),"     1089 inter/extrapolation points."
1585  PRINT TAB(Tab),"  5) This version gets the correlation function from "
1590  PRINT TAB(Tab),"     a fitted formuls."
1595  PRINT TAB(Tab),""
1600  PRINT TAB(Tab),"Send your inquires to"
1605  PRINT TAB(Tab),"     Dr. L. Charles Sun"
1610  PRINT TAB(Tab),"     Deptartment of Oceanography"
1615  PRINT TAB(Tab),"     University of Hawaii"
1620  PRINT TAB(Tab),"     Honolulu, HI 96822"
1625  PRINT TAB(Tab),"      (808)948-7633"
1630  INPUT "Hit [RETURN] or [ENTER] to continue",Answ$
1645  Tx=10
1650  Ty=10
1651  !
1652  Lp_flg=0
1653  LOOP
```

```
1654  OUTPUT KBD;Clear$;
1656  ! ----------------------------------------------------------------
1660  ! Date0$,Time0$ = time of the analysis to make.
1665  PRINT TABXY(Tx,Ty),"Enter time of the analysis to make.(DD MMM YY,HH:MM:SS
)"
1670  PRINT TAB(Ty),"(enclosed with quotations.)"
1671  DISP "Date0$?,Time0$?";
1672  IF Lp_flg THEN OUTPUT KBD;"""";Date0$;""","""";Time0$;"""";
1675  INPUT "",Date0$,Time0$
1680  Date0$=UPC$(Date0$)
1685  OUTPUT KBD;Clear$;
1690  ! ----------------------------------------------------------------
1695  ! Nobj = number of objective analysis to make.
1700  PRINT TABXY(Tx,Ty),"Enter number of objective analysis to make."
1701  DISP "Nobj?";
1702  IF Lp_flg THEN OUTPUT KBD;Nobj;
1705  INPUT "",Nobj
1710  OUTPUT KBD;Clear$;
1715  ! ----------------------------------------------------------------
1720  ! Delta_t = time interval for each extrapolation in time.
1725  PRINT TABXY(Tx,Ty),"Enter time interval for extraploateion in time"
1730  PRINT TAB(Ty),"   or zero for instantaneoue computation."
1731  DISP "Delta_t?";
1732  IF Lp_flg THEN OUTPUT KBD;Delta_t;
1735  INPUT "",Delta_t
1740  OUTPUT KBD;Clear$;
1745  ! ----------------------------------------------------------------
1750  ! Xlimit = Max. distance radius from the reference point of the domain.
1755  PRINT TABXY(Tx,Ty),"Enter Max. distance radius from the reference point of
 the domain."
1760  PRINT TAB(Ty),"All data within this range are retained."
1761  DISP "Xlimit?";
1762  IF Lp_flg THEN OUTPUT KBD;Xlimit;
1765  INPUT "",Xlimit
1770  OUTPUT KBD;Clear$;
1775  ! ----------------------------------------------------------------
1780  ! Tlimit = Max. time radius befor and after the time of the analysis.
1785  PRINT TABXY(Tx,Ty),"Enter the max. time radius befor and after the time of
 the analysis."
1790  PRINT TAB(Ty),"All data within this range are retained."
1791  DISP "Tlimit?";
1792  IF Lp_flg THEN OUTPUT KBD;Tlimit;
1795  INPUT "",Tlimit
1800  OUTPUT KBD;Clear$;
1805  ! ----------------------------------------------------------------
1810  ! Max_space_lag = maximum space lag.
1815  ! Max_time_lag = maximum time lag.
1820  PRINT TABXY(Tx,Ty),"Set the influence domain for computation points,"
1825  PRINT TAB(Ty),"the maximum time lags >= the multiplication of time interva
l"
```

```
1830  PRINT TAB(Ty),"        and the number of the analysis to make."
1835  PRINT TAB(Ty),"the maximum spae lags <= the sizes of the computation domai
n."
1840  PRINT
1845  PRINT TAB(Ty),"Enter the maximum space and time lags."
1846  DISP "Max_space_lag?, Max_time_lag?";
1847  IF Lp_flg THEN OUTPUT KBD;Max_space_lag,Mag_time_lag;
1850  INPUT "",Max_space_lag,Max_time_lag
1855  OUTPUT KBD;Clear$;
1860  ! -----------------------------------------------------------------
1865  ! Limit = Max. number of influential points.
1870  PRINT TABXY(Tx,Ty),"Enter the maximum number of influential points."
1871  DISP "Limit?";
1872  IF Lp_flg THEN OUTPUT KBD;Limit;
1875  INPUT "",Limit
1880  OUTPUT KBD;Clear$;
1881  ! -----------------------------------------------------------------
1882  ! Xfold = X direction e-folding (0.36) scale for the correlation function
1883  PRINT TABXY(Tx,Ty),"Enter the X direction E-folding scale"
1884  DISP "Xfold?";
1885  IF Lp_flg THEN OUTPUT KBD;Xfold;
1886  INPUT "",Xfold
1887  OUTPUT KBD;Clear$;
1889  ! -----------------------------------------------------------------
1890  ! Yfold = Y direction e-folding (0.36) scale for the correlation function
1891  PRINT TABXY(Tx,Ty),"Enter the Y direction E-folding scale"
1892  DISP "Yfold?";
1893  IF Lp_flg THEN OUTPUT KBD;Yfold;
1894  INPUT "",Yfold
1895  OUTPUT KBD;Clear$;
1896  ! -----------------------------------------------------------------
1897  ! Tfold = Time (SECONDS) e-folding (0.36) scale for the correlation functi
on
1898  PRINT TABXY(Tx,Ty),"Enter the TIME e-folding scale (SECONDS)"
1899  DISP "Tfold?";
1900  IF Lp_flg THEN OUTPUT KBD;Tfold;
1901  INPUT "",Tfold
1902  OUTPUT KBD;Clear$;
1904  ! -----------------------------------------------------------------
1905  ! Data_file$, Nskip = Observed data and number of data to be skipped.
1906  PRINT TABXY(Tx,Ty),"Enter the observed data file including file specifiers
."
1907  DISP "Data_file$?";
1908  IF Lp_flg THEN OUTPUT KBD;Data_file$;
1910  INPUT "",Data_file$
1911  Data_file$=UPC$(Data_file$)
1912  OUTPUT KBD;Clear$;
1915  ! -----------------------------------------------------------------
1920  ! Ip_file$ = Interpolation position  data.
1925  PRINT TABXY(Tx,Ty),"Enter the interpolation position file including file s
```

```
pecifiers."
1926  DISP "Ip_file$?";
1927  IF Lp_flg THEN OUTPUT KBD;Ip_file$;
1930  INPUT "",Ip_file$
1935  Ip_file$=UPC$(Ip_file$)
1940  OUTPUT KBD;Clear$;
1945  ! ------------------------------------------------------------------------
1950  ! Voa_fcst$ = VOA forecast field file.
1955  PRINT TABXY(Tx,Ty),"Enter the file specifier for OA forecast fields."
1956  DISP "Voa_fcst$?";
1957  IF Lp_flg THEN OUTPUT KBD;Voa_fcst$;
1960  INPUT "",Voa_fcst$
1965  Voa_fcst$=UPC$(Voa_fcst$)
1970  OUTPUT KBD;Clear$;
1975  ! ------------------------------------------------------------------------
1980  ! Voa_evar$ = VOA expected error file.
1985  PRINT TABXY(Tx,Ty),"Enter the file specifier for OA expected error fields.
"
1986  DISP "Voa_evar$?";
1987  IF Lp_flg THEN OUTPUT KBD;Voa_evar$;
1990  INPUT "",Voa_evar$
1995  Voa_evar$=UPC$(Voa_evar$)
2000  OUTPUT KBD;Clear$;
2005  ! ------------------------------------------------------------------------
2010  !
2015  ! --- Echo check
2020  !
2025  Tab=10
2030  PRINT TABXY(10,2)
2035  PRINT TAB(Tab),"************   Echo check of input variables   ***********
*"
2040  PRINT TAB(Tab),"Time of the analysis: ",Date0$," ",Time0$
2045  PRINT TAB(Tab),"Number of objective analysis to make: ",Nobj
2050  PRINT TAB(Tab),"Time interval:                          ",Delta_t
2055  PRINT TAB(Tab),"Max. distance radius:                   ",Xlimit
2060  PRINT TAB(Tab),"Max. time radius:                       ",Tlimit
2065  PRINT TAB(Tab),"Maximum time lag:                       ",Max_time_lag
2070  PRINT TAB(Tab),"Maximum space lag:                      ",Max_space_lag
2075  PRINT TAB(Tab),"Maximum number of influential points: ",Limit
2080  PRINT TAB(Tab),"The observed data file:                 ",Data_file$
2085  PRINT TAB(Tab),"The interpolated positions file:        ",Ip_file$
2090  PRINT TAB(Tab),"The VOA forecast output file:           ",Voa_fcst$
2095  PRINT TAB(Tab),"The VOA expected error output file:     ",Voa_evar$
2100  Lp_flg=1
2101  Answ$=""
2102  INPUT "Do you wnat to change any values (Y/N)  <no>",Answ$
2103  EXIT IF UPC$(Answ$[1;1])<>"Y"
2104  END LOOP
2115  OUTPUT KBD;Clear$;
2120  !
```

```
2125  ! input the observed data
2130  !
2135  Get_data(Data_file$,Xdata(*),Ydata(*),Tdata(*),Udata(*),Vdata(*),Ndata)
2140  IF Ndata=0 THEN
2145   PRINT "Error in reading the observed data"
2150   STOP
2155  END IF
2160  PRINT "Total number of the observed data: ",Ndata
2165  !
2170  ! read the interpolated positions
2175  !
2180  OUTPUT KBD;Clear$;
2185  Get_ip(Ip_file$,Xip(*),Yip(*),Nip)
2190  !
2195  IF Nip=0 THEN
2200   PRINT "Error in reading interpolated positon, Program stopped"
2205   STOP
2210  END IF
2215  PRINT "Total number of the interpolated positions:",Nip
2220  !
2225  T0=DATE(Date0$)+TIME(Time0$) ! Convert the real time to HP time format
2230  !
2235  ! Loop for each analysis
2240  !
2245  FOR Iobj=1 TO Nobj
2250   Tf=T0+Delta_t*Iobj
2255   Tl=Tf-Tlimit
2260   Tu=Tf+Tlimit
2265   PRINT "Forecast time: ",DATE$(Tf),TIME$(Tf)
2270   E1=0.
2275   E2=0.
2280  !
2285  ! get the observed data points within the limited range and at the proper
time
2290  !
2295   CALL Get_rd(Xdata(*),Ydata(*),Tdata(*),Udata(*),Vdata(*),Ndata,Tl,Tu,Xlim
it,Xrd(*),Yrd(*),Trd(*),Urd(*),Vrd(*),N)
2300   PRINT "Number of data to use in OA: ",N
2305  !
2310   CALL Diag(Urd(*),N)
2315   CALL Diag(Vrd(*),N)
2320  !
2325   IF E1>1.0 THEN
2330    PRINT "The error noise level 100% exceeded; ERROR variance=",E1
2335    STOP
2340   END IF
2345  !
2350  ! Do the analysis for each point
2355  !
2360   FOR Ip=1 TO Nip
```

```
2365    CALL Select(Limit,Xip(Ip),Yip(Ip),Tf,Xrd(*),Yrd(*),Trd(*),Urd(*),Vrd(*),
Xopd(*),Yopd(*),Topd(*),Uopd(*),Vopd(*),N,Nobs,Max_space_lag,Max_time_lag)
2370    CALL Vector_oa(Xopd(*),Yopd(*),Topd(*),Uopd(*),Vopd(*),Nobs,Xip(Ip),Yip(
Ip),Tf,Ues,Ves,Uerr,Verr,Ier)
2375   !
2380    IF Ier=0 THEN 2400
2385    E1=E1+.01
2390    E2=E2+.01
2395    GOTO 2325
2400    Uest(Ip)=Ues
2405    Vest(Ip)=Ves
2410    Uerror(Ip)=Uerr
2415    Verror(Ip)=Verr
2420    DISP USING "K,DDDD,K,MD.DDDE,K,MD.DDDE";"Doing V.O.A. at point no. ",Ip,
"    Est. U: ",Uest(Ip)," Expected Error:",Uerror(Ip)
2425   NEXT Ip
2430   !
2435   ! compute the mean and variance of the fields
2440   !
2445   PRINT
2450   PRINT "the diagnostics of the forecast fields"
2455   CALL Diag(Uest(*),Nip)
2460   CALL Diag(Vest(*),Nip)
2465   PRINT
2470   PRINT "the diagnostics of the error fields"
2475   CALL Diag(Uerror(*),Nip)
2480   CALL Diag(Verror(*),Nip)
2485 !
2490 ! output the V.O.A. forcast field.
2495 !
2500   OUTPUT @Fcst_out;"V.O.A. forecast field"
2505   OUTPUT @Fcst_out;"Time: "&DATE$(Tf)&" "&TIME$(Tf)
2510   DISP "Output the VOA forecast filed to ",Voa_fcst$
2515   FOR Ip=1 TO Nip
2520    OUTPUT @Fcst_out;Uest(Ip),Vest(Ip)
2525   NEXT Ip
2530 !
2535 ! output the V.O.A. error variance field.
2540 !
2545   OUTPUT @Evar_out;"Expected Error field"
2550   OUTPUT @Evar_out;"Time: "&DATE$(Tf)&" "&TIME$(Tf)
2555   DISP "Output the VOA error variance filed to ",Voa_evar$
2560   FOR Ip=1 TO Nip
2565    OUTPUT @Evar_out;Uerror(Ip),Verror(Ip)
2570   NEXT Ip
2575    !
2580   DISP
2585  NEXT Iobj
2590   !
2595   ! close all files
```

```
2600   !
2605  ASSIGN @Fcst_out TO *
2610  ASSIGN @Evar_out TO *
2615  Cpu_time=TIMEDATE-DATE(Sdate$)-TIME(Stime$)
2620  PRINT "CUP time for this run = ",Cpu_time
2625  END
2630  !
2635  !------------------------------------------------------------------
2640  !
2645  SUB Get_rd(Xdata(*),Ydata(*),Tdata(*),Udata(*),Vdata(*),M,Tl,Tu,Xlimit,Xrd
(*),Yrd(*),Trd(*),Urd(*),Vrd(*),N)
2650  !
2655  ! --- routine gets the data before and after a given time and within
2660  !      a given spatial radius from the domain reference point.
2665  !
2670   RAD
2675   OPTION BASE 1
2680   IF Tl=Tu THEN
2685    PRINT USING "K,K,K";"Use data on ",DATE$(Tu)," ",TIME$(Tu)
2690   ELSE
2695    PRINT USING "K,K,K,K,K,K,K,K";"Use data before ",DATE$(Tu)," ",TIME$(Tu)
," and after ",DATE$(Tl)," ",TIME$(Tl)
2700   END IF
2705   PRINT USING "K,MD.DDDE,K";" and within thin the range of ",Xlimit," from
the reference point."
2710   N=0
2715   FOR I=1 TO M
2720    IF Tdata(I)>Tu THEN GOTO Next_i
2725    IF Tdata(I)<Tl THEN GOTO Next_i
2730    IF ABS(Xdata(I))>Xlimit THEN Next_i
2735    IF ABS(Ydata(I))>Xlimit THEN Next_i
2740    N=N+1
2745    Xrd(N)=Xdata(I)
2750    Yrd(N)=Ydata(I)
2755    Trd(N)=Tdata(I)
2760    Urd(N)=Udata(I)
2765    Vrd(N)=Vdata(I)
2770 Next_i:NEXT I
2775   IF N=0 THEN
2780    PRINT "No data were found in this range, please re-define time or space
ranges."
2785    PRINT "Program execution halted."
2790    BEEP
2795    STOP
2800   ELSE
2805    PRINT "No. of data within this range: ",N
2810   END IF
2815  SUBEND
2820  !
2825  ! -----------------------------------------------------------------
```

```
2830  !
2835  SUB De_mean(Phird(*),M,Mean,Sdv)
2840   OPTION BASE 1
2845   RAD
2850  !
2855  ! routine calculates the meAN and standard deviation of an array
2860  ! it also removes the mean from the array
2865  !
2870   Mean=0.
2875   Sdv=0.
2880   FOR I=1 TO M
2885    Mean=Mean+Phird(I)
2890    Sdv=Sdv+Phird(I)^2
2895   NEXT I
2900   Mean=Mean/M
2905   Sdv=Sdv/M-Mean^2
2910   IF M<=1 THEN Sdv=(M/(M-1.))*Sdv
2915   FOR I=1 TO M
2920    Phird(I)=Phird(I)-Mean
2925   NEXT I
2930  SUBEND
2935  !
2940  ! -------------------------------------------------------------------
2945  !
2950  SUB Select(Limit,X,Y,Tf,Xrd(*),Yrd(*),Trd(*),Urd(*),Vrd(*),Xopd(*),Yopd(*)
,Topd(*),Uopd(*),Vopd(*),N,Nobs,Max_space_lag,Max_time_lag)
2955   OPTION BASE 1
2960   RAD
2965  !
2970  ! --- routine eliminates the distant (in space and time ) points and
2975  !     and selects the most "LIMIT" near points to an interpolation
2980  !     point X,Y,Tf.
2985  !
2990   DIM Index(1089),Error(1089)
2995   COM /Efield/ E1,E2
3000   Cphase=0.
3005   Nobs=0
3010   R=FNUu(0.,0.,0.)*FNVv(0.,0.,0.)-FNUv(0.,0.,0.)^2
3015   A11=FNVv(0.,0.,0.)/R
3020   A22=FNUu(0.,0.,0.)/R
3025   A12=FNUv(0.,0.,0.)/R
3030   FOR I=1 TO N
3035    Delt=Tf-Trd(I)
3040    IF ABS(Delt)>Max_time_lag THEN 3105
3045    Delx=X-Xrd(I)
3050    Dely=Y-Yrd(I)
3055    R=SQR((Delx-Cphase*Delt)^2+Dely^2)
3060    IF R>Max_space_lag THEN 3105
3065    Nobs=Nobs+1
3070    Index(Nobs)=I
```

```
3075     Cx1=FNUu(Delx,Dely,Delt)
3080     Cx2=FNUv(Delx,Dely,Delt)
3085     Cy1=Cx2
3090     Cy2=FNVv(Delx,Dely,Delt)
3095     Error(Nobs)=(Cx1^2+Cy1^2)*A11+2.*(Cx1*Cx2+Cy1*Cy2)*A12+(Cx2^2+Cy2^2)*A22
3100     Error(Nobs)=ABS(2.-Error(Nobs))
3105    NEXT I
3110    IF Nobs=0 THEN 3175
3115    IF Nobs>Limit THEN
3120     CALL Sort(Error(*),Index(*),Nobs)
3125     Nobs=Limit
3130    END IF
3135    FOR I=1 TO Nobs
3140     J=Index(I)
3145     Xopd(I)=Xrd(J)
3150     Yopd(I)=Yrd(J)
3155     Topd(I)=Trd(J)
3160     Uopd(I)=Urd(J)
3165     Vopd(I)=Vrd(J)
3170    NEXT I
3175   SUBEND
3180   !
3185   ! ------------------------------------------------------------------
3190   !
3195   SUB Sort(Cor(*),Index(*),N)
3200   !
3205   ! --- A shell sort routine to sort index and cor up according to
3210   !       the values of cor
3215   !
3220    OPTION BASE 1
3225    RAD
3230    Igap=N
3235    IF Igap<=1 THEN 3320
3240    Igap=Igap/2
3245    Imax=N-Igap
3250    Iex=0
3255    FOR I=1 TO Imax
3260     Iplusg=I+Igap
3265     IF Cor(I)>=Cor(Iplusg) THEN 3305
3270     Save=Cor(I)
3275     Cor(I)=Cor(Iplusg)
3280     Cor(Iplusg)=Save
3285     Isave=Index(I)
3290     Index(I)=Index(Iplusg)
3295     Index(Iplusg)=Isave
3300     Iex=1
3305    NEXT I
3310    IF Iex<>0 THEN 3250
3315    GOTO 3235
3320   SUBEND
```

```
3325  !
3330  !------------------------------------------------------------------------
3335  !
3340  SUB Vector_oa(Xopd(*),Yopd(*),Topd(*),Uopd(*),Vopd(*),N,X,Y,Tf,Uest,Vest,Uerr,Verr,Ier)
3345  !
3350  ! --- The vector space-time objective analysis routine.
3355  !
3360   OPTION BASE 1
3365   RAD
3370   DIM A(80,80),Cu(2178),Cv(2178)
3375   COM /Efield/ E1,E2
3380   Uest=0.
3385   Vest=0.
3390   Uerr=1.
3395   Verr=1.
3400   Ier=-1
3405   IF N<=0 THEN GOTO 3615
3410   CALL Set_inva(A(*),Xopd(*),Yopd(*),Topd(*),N,Ier)
3415   IF Ier>0 THEN 3615
3420   !
3425   ! --- calculate the correlation matricies CU and CV.
3430   !
3435   FOR J=1 TO N
3440    Delt=Tf-Topd(J)
3445    Delx=X-Xopd(J)
3450    Dely=Y-Yopd(J)
3455    Cu(J)=FNUu(Delx,Dely,Delt)
3460    Cu(J+N)=FNUv(Delx,Dely,Delt)
3465    Cv(J+N)=FNVv(Delx,Dely,Delt)
3470    Cv(J)=Cu(J+N)
3475   NEXT J
3480   !
3485   ! --- calculate the error variances Uerr and Verr and velocity
3490   !      estimates, Uest and Vest.
3495   !
3500   N2=N*2
3505   Uerr=0.
3510   Verr=0.
3515   FOR I=1 TO N2
3520    Ipn=I+N
3525    Eta=0.
3530    FOR J=1 TO N
3535     Jpn=J+N
3540     Eta=Eta+A(I,J)*Uopd(J)+A(I,Jpn)*Vopd(J)
3545     IF I<=N THEN
3550      Uerr=Uerr+Cu(I)*A(I,J)*Cu(J)+Cu(Ipn)*A(I,Jpn)*Cv(J)
3555      Verr=Verr+Cv(Ipn)*A(Ipn,Jpn)*Cv(Jpn)+Cu(Ipn)*A(I,Jpn)*Cv(J)
3560     END IF
3565    NEXT J
```

```
3570     Uest=Uest+Cu(I)*Eta
3575     Vest=Vest+Cv(I)*Eta
3580    NEXT I
3585    Uest=Uest+Uave
3590    Vest=Vest+Vave
3595    Cxxu=FNUu(0.,0.,0.)+FNUv(0.,0.,0.)
3600    Cxxv=FNUv(0.,0.,0.)+FNVv(0.,0.,0.)
3605    Uerr=1.-Uerr/Cxxu
3610    Verr=1.-Verr/Cxxv
3615   SUBEND
3620   !
3625   !------------------------------------------------------------------------
3630   !
3635   SUB Set_inva(A(*),Xopd(*),Yopd(*),Topd(*),Nobs,Ier)
3640   !
3645   ! --- routine sets up the correlation function for the observations
3650   !     given the positions Xopd, Yopd and times, Topd, it returns the
3655   !     inverted correlation function matrix.
3660   !
3665    OPTION BASE 1
3670    RAD
3675    DIM Ip(1089)
3680    COM /Efield/ E1,E2
3685    Guard=1.0E-30
3690    D1=1.
3695    Ier=0
3700    FOR I=1 TO Nobs
3705     Ipnobs=I+Nobs
3710     FOR J=I TO Nobs
3715      Jpnobs=J+Nobs
3720      Delt=Topd(I)-Topd(J)
3725      Delx=Xopd(I)-Xopd(J)
3730      Dely=Yopd(I)-Yopd(J)
3735      A(I,J)=FNUu(Delx,Dely,Delt)
3740      A(I,Jpnobs)=FNUv(Delx,Dely,Delt)
3745      A(Ipnobs,J)=A(I,Jpnobs)
3750      A(Ipnobs,Jpnobs)=FNVv(Delx,Dely,Delt)
3755     NEXT J
3760     A(I,I)=A(I,I)+E1
3765     A(Ipnobs,Ipnobs)=A(Ipnobs,Ipnobs)+E2
3770    NEXT I
3775    N2=Nobs*2
3780   !
3785   ! --- invert the (nobs*2)*(nobs*2) matrix A
3790   !
3795    CALL Invmtx(A(*),N2,A(*),N2,N2,D,Ip(*),Ier)
3800    IF D<Guard THEN
3805     PRINT "Warning the determinant is very small; DET= ";D
3810     Ier=-1
3815    END IF
```

```
3820  SUBEND
3825  !
3830  !------------------------------------------------------------------
3835  !
3840  SUB Est_mean(A(*),Xopd(*),Yopd(*),Topd(*),N,U(*),V(*),Uave,Vave)
3845  !
3850  ! --- routine calculates the estimated mean, and then removes
3855  !     the estimated mean from the observations.
3860  !
3865   OPTION BASE 1
3870   RAD
3875   DIM C(2178),D(2178)
3880   CALL Set_inva(A(*),Xopd(*),Yopd(*),Topd(*),N,Ier)
3885   N2=N*2
3890   FOR I=1 TO N2
3895    Ipn=I+P
3900    C(I)=0.
3905    D(I)=0.
3910    FOR K=1 TO N
3915     Kpn=K+N
3920     C(I)=C(I)+A(I,K)*U(K)+A(I,Kpn)*V(K)
3925     D(I)=D(I)+A(I,K)+A(I,Kpn)
3930    NEXT K
3935   NEXT I
3940   Sum1=0.
3945   Sum2=0.
3950   Sum3=0.
3955   Sum4=0.
3960   FOR I=1 TO N
3965    Ipn=I+N
3970    Sum1=Sum1+C(I)
3975    Sum2=Sum2+D(I)
3980    Sum3=Sum3+C(Ipn)
3985    Sum4=Sum4+D(Ipn)
3990   NEXT I
3995   Uave=Sum1/Sum2
4000   Vave=Sum3/Sum4
4005   !
4010   ! --- remove the calculated means
4015   !
4020   FOR I=1 TO N
4025    U(I)=U(I)-Uave
4030    V(I)=V(I)-Vave
4035   NEXT I
4040  SUBEND
4045  !
4050  !------------------------------------------------------------------
4055  !
4060  SUB Diag(Opsi(*),M)
4065   OPTION BASE 1
```

```
4070  RAD
4075  !
4080  IF M<=0 THEN 4190
4085  PRINT
4090  PRINT "Number of points: ",M
4095  Ave=0.
4100  Sdv=0.
4105  Psimax=Dpsi(1)
4110  Psimin=Dpsi(1)
4115  Ave=Dpsi(1)
4120  IF M=1 THEN 4155
4125  FOR I=2 TO M
4130   IF Psimax<Dpsi(I) THEN Psimax=Dpsi(I)
4135   IF Psimin>Dpsi(I) THEN Psimin=Dpsi(I)
4140   Sdv=((I-2)*Sdv+(I-1)*(Dpsi(I)-Aver)^2/I)/(I-1).
4145   Ave=((I-1)*Ave+Dpsi(I))/I
4150  NEXT I
4155  Rms=SQR(Sdv+Ave^2)
4160  PRINT "Mean and Variance: ",Ave,Sdv
4165  Sdv=SQR(Sdv)
4170  PRINT "Standard Deviation: ",Sdv
4175  PRINT "RMS of field: ",Rms
4180  PRINT "minimum of field: ",Psimin
4185  PRINT "maximum of field: ",Psimax
4190 SUBEND
4195 !
4200 ! ------------------------------------------------------------------
4205 !
4210 SUB Invmtx(A(*),Na,V(*),Nv,N,D,Ip(*),Ier)
4215 !
4220 ! --- routine inverts the Matrix A.
4225 !
4230  OPTION BASE 1
4235  RAD
4240  Iexmax=75
4245  Ier=FNIerinv(N,Na,Nv)
4250  IF Ier<>0 THEN 4600
4255  FOR J=1 TO N
4260   Ip(J)=0
4265   FOR I=1 TO N
4270    V(I,J)=A(I,J)
4275   NEXT I
4280  NEXT J
4285  D=1.
4290  Iex=0
4295  FOR M=1 TO N
4300   Vmax=0.
4305   FOR J=1 TO N
4310    IF Ip(J)<>0 THEN 4355
4315    FOR I=1 TO N
```

```
4320      IF Ip(I)<>0 THEN 4350
4325      Vh=ABS(V(I,J))
4330      IF Vmax>Vh THEN 4350
4335      Vmax=Vh
4340      K=I
4345      L=J
4350     NEXT I
4355    NEXT J
4360    Ip(L)=K
4365    Npm=N+M
4370    Ip(Npm)=L
4375    D=D*V(K,L)
4380    IF ABS(D)<=1.0 THEN 4400
4385    D=D*.1
4390    Iex=Iex+1
4395    GOTO 4380
4400    Pvt=V(K,L)
4405    IF M=1 THEN Pvtmx=ABS(Pvt)
4410    IF (ABS(Pvt/M)+Pvtmx)=Pvtmx THEN 4570
4415    V(K,L)=1.
4420    FOR J=1 TO N
4425     Hold=V(K,J)
4430     V(K,J)=V(L,J)
4435     V(L,J)=Hold/Pvt
4440    NEXT J
4445    FOR I=1 TO N
4450     IF I=L THEN 4480
4455     Hold=V(I,L)
4460     V(I,L)=0.
4465     FOR J=1 TO N
4470      V(I,J)=V(I,J)-V(L,J)*Hold
4475     NEXT J
4480    NEXT I
4485   NEXT M
4490   M=N+N+1
4495   FOR J=1 TO N
4500    M=M-1
4505    L=Ip(M)
4510    K=Ip(L)
4515    IF K=L THEN 4550
4520    D=-D
4525    FOR I=1 TO N
4530     Hold=V(I,L)
4535     V(I,L)=V(I,K)
4540     V(I,K)=Hold
4545    NEXT I
4550   NEXT J
4555   IF Iex>Iexmax THEN 4585
4560   D=D*10.^Iex
4565   GOTO 4600
```

```
4570   Ier=33
4575   PRINT "MATRIX SINGULAR IN INVMTX; ERROR CODE IS ",Ier
4580   GOTO 4600
4585   Ier=1
4590   D=Iex
4595   PRINT "DETERMINANT TOO LARGE IN INVMTX; ERROR CODE IS ",Ier
4600  SUBEND
4605  !
4610  ! -------------------------------------------------------------------
4615  !
4620  DEF FNIerinv(N,Na,Nv)
4625   OPTION BASE 1
4630   RAD
4635   Ierinv=0
4640   IF N>=1 THEN 4660
4645   Ierinv=34
4650   PRINT "N < 1 IN INVMTX"
4655   GOTO 4695
4660   IF Na>=N THEN 4680
4665   Ierinv=35
4670   PRINT "NA<N IN INVMTX"
4675   GOTO 4695
4680   IF Nv>=N THEN 4695
4685   Ierinv=36
4690   PRINT "NV < N IN INVMTX"
4695   RETURN Ierinv
4700  FNEND
4705  !
4710  !
4715  !
4720 Get_data:SUB Get_data(Filename$,Xdata(*),Ydata(*),Tdata(*),Udata(*),Vdata(*
),Ndata)
4725  !
4730  ! --- routine reads in the observations.
4735  !
4740   ASSIGN @Path_in TO Filename$
4745   Ndata=0
4750   PRINT "      **********   Echo check of the first ten records   **********
*"
4755   PRINT "  No.      X_POS          Y_POS             TIME          U-COMP.
 V-COMP."
4760   ON END @Path_in GOTO 4815
4765   DISP USING "K,K,K";"Reading the observed data from ",Filename$,", please
wait."
4766  LOOP
4770   ENTER @Path_in;X,Y,T,U,V
4775   IF Ndata<=10 THEN PRINT USING "DDDD,X,MD.DDDDDDE,X,MD.DDDDDDE,X,MD.DDDDDD
DDDE,X,MD.DDDDDDE,X,MD.DDDDDDE";Ndata,X,Y,T,U,V
4780   Ndata=Ndata+1
4785   Xdata(Ndata)=X
```

```
4790   Ydata(Ndata)=Y
4795   Tdata(Ndata)=T
4800   Udata(Ndata)=U
4805   Vdata(Ndata)=V
4810  END LOOP
4815   ASSIGN @Path_in TO *! Closing input_file.
4820   DISP
4825  SUBEND
4830  !
4835  !
4840  !
4845 Get_data:SUB Get_ip(Filename$,Xdata(*),Ydata(*),Nip)
4850  !
4855  ! --- routine reads in the inter/extrapolation positions.
4860  !
4865   ASSIGN @Path_in TO Filename$
4870   PRINT "* Echo check of the first ten records *"
4875   PRINT "  No.    X_IP_POS.       Y_IP_POS."
4880   Nip=0
4885   ON END @Path_in GOTO Close_file
4890   DISP USING "K,K,K";"Reading the interpolated positions from ",Filename$,", please wait."
4891  LOOP
4895   ENTER @Path_in;X,Y
4900   IF Nip<=10 THEN PRINT USING "DDDD,XX,MD.DDDDDDE,XX,MD.DDDDDDE";Nip,X,Y
4905   Nip=Nip+1
4910   Xdata(Nip)=X
4915   Ydata(Nip)=Y
4920  END LOOP
4925 Close_file:ASSIGN @Path_in TO *  ! Closing input_file.
4930   DISP
4935  SUBEND
4940  !
```

```
10     ! PROGRAM SOA_DIFF
11     !
20     ! ---  Program Calculates Diference Between Two 15x15 Arrays
30     !
1000  OPTION BASE 1
1005  DIM File$[40],Title1$[80],Title2$[80]
1010  REAL Farea1(15,15),Farea2(15,15),Farea(15,15),Farea_sum
1015  !
1020  OUTPUT KBD;CHR$(255)&CHR$(75);
1025  File$=""
1030  LINPUT "Enter first file to compute difference (include device)  <exit>",F
ile$
1035  File$=TRIM$(UPC$(File$))
1040  IF File$="" THEN GOTO Stop_program
1045  ON ERROR GOTO Data_error
1050  ASSIGN @File TO File$
1055  ENTER @File;Title1$,Title2$,Farea1(*)
1060  ASSIGN @File TO *
1065  !
1070  File$=""
1075  LINPUT "Enter second file to compute difference (include device)  <exit>",
File$
1080  File$=TRIM$(UPC$(File$))
1085  IF File$="" THEN GOTO Stop_program
1090  ASSIGN @File TO File$
1095  ENTER @File;Title1$,Title2$,Farea2(*)
1100  ASSIGN @File TO *
1110  File$=""
1115  LINPUT "Enter file for File1 - File2 difference (include device)  <exit>",
File$
1120  File$=TRIM$(UPC$(File$))
1125  IF File$="" THEN GOTO Stop_program
1130  ON ERROR GOTO 1140
1135  CREATE BDAT File$,10
1140  ON ERROR GOTO Data_error
1145  ASSIGN @File TO File$
1150  MAT Farea= Farea1-Farea2
1155  OUTPUT @File;Title1$,Title2$,Farea(*)
1160  ASSIGN @File TO *
1165  MAT Farea= Farea . Farea
1170  Farea_sum=SQR(SUM(Farea))
1175  DISP "Cummulated Difference:";Farea_sum
1180  ASSIGN @File TO *
1185  STOP
1190 Stop_program: !
1195  DISP "Stopped"
1200  STOP
1205 Data_error: !
1210  DISP ERRM$;"; File '";File$;"'"
1215  END
```

```
1000  !PROGRAM PLOT_OA
1005  !
1010  ! --- Procram plots the objective analysis field.
1015  !
1020  OPTION BASE 1
1025  DIM Title1$[80],Title2$[80],User$[40],Answ$[10],Old$[40],Oa_file$[40]
1030  DIM Sfc(15,15)
1035  ! ---------------------------------------------------------------
1040  ! Assign the Objective analysis results file to Oa_file$
1045  Old$=""
1050  LOOP
1055   OUTPUT KBD;CHR$(255)&CHR$(75);
1060   LINPUT "Enter the O.A. data file to be plotted  <exit>",Oa_file$
1065   Oa_file$=TRIM$(UPC$(Oa_file$))
1070  EXIT IF Oa_file$=""
1075  ! ---------------------------------------------------------------
1080  ! Input the x- and y-grid points of computating field.
1085   INPUT "Enter the x- and y-grid points of computating field, (Imax,Jmax)",
Imax,Jmax
1090   IF Imax<=21 OR Jmax<=21 THEN 1130
1095   BEEP
1100   PRINT TABXY(10,10),"                    *** Warning ***"
1105   PRINT TABXY(10,11),"The sizes of array Sfc(i,j) defined in this version"
1110   PRINT TABXY(10,12),"is incorrect, make correction in lines 1015 and 1055.
"
1115   PRINT TABXY(10,13),"Program execution is halted."
1120   STOP
1125  ! Set the minimum and maximum contours
1130   INPUT "Enter the minimum contour to be plotted.",Min
1135   INPUT "Enter the Maximum contour to be plotted.",Max
1140  ! ---------------------------------------------------------------
1145  ! Set number of contours(Nc) or set  contour interval(Ci), if Nc=0
1150   INPUT "Enter the number of contour or the Contour interval",Nc
1155   IF INT(Nc*100/100)=Nc THEN
1160    Ci=ABS(Max-Min)/Nc
1165   ELSE
1170    Ci=Nc
1175   END IF
1180  ! ---------------------------------------------------------------
1185  ! user label
1190   User$=""
1195   LINPUT "Enter the user's label (40 char max)",User$
1200   User$=TRIM$(User$)
1205  ! ---------------------------------------------------------------
1210  !  screen or plotter
1215   Answ$="S"
1220   INPUT "Plot on screen or pen plotter (S/P)  <screen>",Answ$
1225   Dvc=0
1230   IF UPC$(Answ$[1;1])="P" THEN Dvc=1
1235  ! ---------------------------------------------------------------
```

```
1240  ! dump graphics to printer
1245   Dump_graph=0
1250   IF Dvc=0 THEN
1255    Answ$="N"
1260    INPUT "Dump the graphics to printer (Y/N)  <no>",Answ$
1265    IF UPC$(Answ$[1;1])="Y" THEN Dump_graph=1
1270   END IF
1275  ! -----------------------------------------------------------------
1280   C$=CHR$(255)&"K"
1285  ! -----------------------------------------------------------------
1290  ! read in data
1295   IF Old$<>Oa_file$ THEN
1300    OUTPUT 2 USING "#,K";C$          ! Clean the screen
1305    ASSIGN @Path_in TO Oa_file$
1310    ON END @Path_in GOTO 1505
1315    ENTER @Path_in;Title1$
1320    ENTER @Path_in;Title2$
1325    FOR I=Imax TO 1 STEP -1
1330     FOR J=1 TO Jmax
1335      ON END @Path_in GOTO 1350
1340      ENTER @Path_in;Sfc(I,J)
1345     NEXT J
1350    NEXT I
1355    OFF END @Path_in
1360    ASSIGN @Path_in TO *
1365    Old$=Oa_file$
1370   END IF
1375   OUTPUT 2 USING "#,K";C$           ! Clean the screen
1380   IF Dvc=0 THEN
1385    GINIT
1390    GCLEAR
1395    GRAPHICS ON
1400   ELSE
1405    PLOTTER IS 705,"HPGL"
1410   END IF
1415   Contour(Sfc(*),Min,Max,Ci,1.,Title1$,Title2. U.  ?)! do the plotting
1420   BEEP
1425   IF Dvc=0 THEN
1430    ON KBD ALL GOTO 1440
1435 Idle:GOTO Idle   ! view the plot as long as you want.
1440    OFF KBD
1445    IF Dump_graph THEN
1450     GRAPHICS OFF
1455     DISP "Dumping graphics ..."
1460     DUMP GRAPHICS CRT TO #PRT
1465    END IF
1470    GCLEAR
1475    GRAPHICS OFF
1480   END IF
1485  END LOOP
```

```
1490  DISP "Stopped"
1495  STOP
1500  ! ------------------------------------------------------------------------
1505  OFF END @Path_in
1510  ASSIGN @Path_in TO *
1515  DISP ERRM$;"; File '";Oa_file$;"'"
1520  END
1525  !
1530  !
1535  !
1540 Contour:SUB Contour(Sfc(*),Min,Max,Interval,Extremes,Title1$,Title2$,User$)
1545   OPTION BASE 1
1550  !--------------------------------------------------------------------------!
1555  !                    Copyright 1983, Hewlett-Packard Company               !
1560  !                                All Rights Reserved                       !
1565  !                                                                          !
1570  !    This subprogram plots a contour map of the array Sfc(*), and          !
1575  !  optionally plots local minima, maxima, and statistics.                  !
1580  !                                                                          !
1585  !  Sfc(*):       This is the two-dimensional real array containing the     !
1590  !                data to be plotted.  It need not be square.               !
1595  !  Min & Max:    These are the lowest and highest levels, respectively,    !
1600  !                of the contour lines.  These allow you to specify the     !
1605  !                exact range within which you want contours.  Every        !
1610  !                contour line outside of this range will not be plotted.   !
1615  !  Interval:     This specifies how far apart the contour lines have to    !
1620  !                be (in value, not in distance).  The smaller the inter-   !
1625  !                val, the denser the contour plot.                         !
1630  !  Extremes:     This is a logical variable which specifies whether or     !
1635  !                not to label local maxima and minima.  A local maximum    !
1640  !                is a point whose value is larger that its eight           !
1645  !                neighbors immediately to the west, northwest, north,      !
1650  !                northeast, east, southeast, south, and southwest.  A      !
1655  !                local minimum has a corresponding definition.             !
1660  !--------------------------------------------------------------------------!
1665   INTEGER I,J,Imax,Jmax
1670   COM /G_units/ Gdu_xmax,Gdu_ymax,Udu_xmin,Udu_xmax,Udu_ymin,Udu_ymax,Show
1675   CALL Gdu(X_gdu_max,Y_gdu_max,Xmid,Ymid)
1680   VIEWPORT 0.,X_gdu_max,0,Y_gdu_max
1685   WINDOW 0.,X_gdu_max,0,Y_gdu_max
1690   LINE TYPE 1
1695   CALL Label(4,.6,0,5,1,Xmid,.98*Y_gdu_max,Title1$)
1700   CALL Label(4,.6,0,5,1,Xmid,.94*Y_gdu_max,Title2$)
1705   Min_$=VAL$(INT(MIN(Sfc(*))*1000)/1000)
1710   Max_$=VAL$(INT(MAX(Sfc(*))*1000)/1000)
1715   Ci_$=VAL$(INT(Interval*1000)/1000)
1720   CALL Label(3.5,.6,0,5,1,Xmid,.1*Y_gdu_max,"Min.: "&Min_$&";  Max.: "&Max_
$&";  CI: "&Ci_$)
1725   CALL Label(3.5,.6,0,5,1,Xmid,.05*Y_gdu_max,User$)
1730   VIEWPORT 0.,X_gdu_max,.14*Y_gdu_max,Y_gdu_max*.9
```

```
1735   Imax=SIZE(Sfc,1)
1740   Jmax=SIZE(Sfc,2)
1745   Show(1,(Jmax),(Imax),1)
1750   LINE TYPE 1
1755   PEN 1
1760   CLIP 1,Jmax,Imax,1
1765   AXES 1,1,1,1,1,1,1
1770   AXES 1,1,Jmax,Imax,1,1,1
1775   Northeast=0 !   \
1780   Northwest=1 !    >  Figure what to do for case 4.
1785   Cross=0     !   /
1790   PEN 2
1795   FOR I=1 TO Imax-1
1800    FOR J=1 TO Jmax-1
1805     Big=MAX(Sfc(I,J),Sfc(I,J+1),Sfc(I+1,J),Sfc(I+1,J+1))
1810     Small=MIN(Sfc(I,J),Sfc(I,J+1),Sfc(I+1,J),Sfc(I+1,J+1))
1815     FOR Cont=Min TO Max STEP Interval
1820      IF Cont>Small AND Cont<Big THEN
1825       LINE TYPE 1
1830       IF Cont<0 THEN LINE TYPE 4
1835       Top=Cont>MIN(Sfc(I,J),Sfc(I,J+1)) AND Cont<MAX(Sfc(I,J),Sfc(I,J+1))
1840       Bottom=Cont>MIN(Sfc(I+1,J),Sfc(I+1,J+1)) AND Cont<MAX(Sfc(I+1,J),Sfc(
I+1,J+1))
1845       Left=Cont>MIN(Sfc(I,J),Sfc(I+1,J)) AND Cont<MAX(Sfc(I,J),Sfc(I+1,J))
1850       Right=Cont>MIN(Sfc(I,J+1),Sfc(I+1,J+1)) AND Cont<MAX(Sfc(I,J+1),Sfc(I
+1,J+1))
1855       SELECT Top+Bottom+Left+Right
1860       CASE 0       ! Do nothing..............................................
1865       CASE 2       ! Two intersections, so draw one line..................
1870        IF Top THEN
1875         Jtop=J+(Cont-Sfc(I,J))/(Sfc(I,J+1)-Sfc(I,J))
1880         IF Bottom THEN         ! Top and Bottom............................
1885          Jbottom=J+(Cont-Sfc(I+1,J))/(Sfc(I+1,J+1)-Sfc(I+1,J))
1890          MOVE Jtop,I
1895          DRAW Jbottom,I+1
1900         ELSE          ! (not Bottom)
1905          IF Left THEN          ! Top and Left..............................
1910           Ileft=I+(Cont-Sfc(I,J))/(Sfc(I+1,J)-Sfc(I,J))
1915           MOVE Jtop,I
1920           DRAW J,Ileft
1925          ELSE          ! Not left, therefore Top and Right................
1930           Iright=I+(Cont-Sfc(I,J+1))/(Sfc(I+1,J+1)-Sfc(I,J+1))
1935           MOVE Jtop,I
1940           DRAW J+1,Iright
1945          END IF          ! (if left)
1950         END IF          ! (if bottom)
1955        ELSE         ! (not Top)
1960         IF Bottom THEN
1965          Jbottom=J+(Cont-Sfc(I+1,J))/(Sfc(I+1,J+1)-Sfc(I+1,J))
1970          IF Left THEN          ! Bottom and Left..........................
```

```
1975           Ileft=I+(Cont-Sfc(I,J))/(Sfc(I+1,J)-Sfc(I,J))
1980           MOVE J,Ileft
1985           DRAW Jbottom,I+1
1990          ELSE           ! Not left, therefore Bottom and Right..............
1995           Iright=I+(Cont-Sfc(I,J+1))/(Sfc(I+1,J+1)-Sfc(I,J+1))
2000           MOVE Jbottom,I+1
2005           DRAW J+1,Iright
2010          END IF           ! (if left)
2015         ELSE           ! Not Bottom, therefore Left and Right................
2020          Ileft=I+(Cont-Sfc(I,J))/(Sfc(I+1,J)-Sfc(I,J))
2025          Iright=I+(Cont-Sfc(I,J+1))/(Sfc(I+1,J+1)-Sfc(I,J+1))
2030          MOVE J,Ileft
2035          DRAW J+1,Iright
2040         END IF           ! (if bottom)
2045        END IF          ! (if top)
2050       CASE 4         ! Four intersections...................................
2055        Jtop=J+(Cont-Sfc(I,J))/(Sfc(I,J+1)-Sfc(I,J))
2060        Jbottom=J+(Cont-Sfc(I+1,J))/(Sfc(I+1,J+1)-Sfc(I+1,J))
2065        Ileft=I+(Cont-Sfc(I,J))/(Sfc(I+1,J)-Sfc(I,J))
2070        Iright=I+(Cont-Sfc(I,J+1))/(Sfc(I+1,J+1)-Sfc(I,J+1))
2075        IF Northeast THEN
2080         MOVE J,Ileft
2085         DRAW Jtop,I
2090         MOVE Jbottom,I+1
2095         DRAW J+1,Iright
2100        END IF          ! (if northeast)
2105        IF Northwest THEN
2110         MOVE J,Ileft
2115         DRAW Jbottom,I+1
2120         MOVE Jtop,I
2125         DRAW J+1,Iright
2130        END IF          ! (if northwest)
2135        IF Cross THEN
2140         MOVE J,Ileft
2145         DRAW J+1,Iright
2150         MOVE Jtop,I
2155         DRAW Jbottom,I+1
2160        END IF          ! (if cross)
2165       END SELECT
2170      END IF
2175     NEXT Cont
2180    NEXT J
2185   NEXT I
2190   IF Extremes>0 THEN
2195   !
2200    LINE TYPE 1
2205    Image$="K"
2210    FOR I=2 TO Imax-1
2215     FOR J=2 TO Jmax-1
2220      Point=Sfc(I,J)    ! (The point we're working on)
```

```
2225      Min=MIN(Sfc(I-1,J-1),Sfc(I-1,J),Sfc(I-1,J+1),Sfc(I,J-1),Sfc(I,J+1),Sfc
(I+1,J-1),Sfc(I+1,J),Sfc(I+1,J+1))
2230      Max=MAX(Sfc(I-1,J-1),Sfc(I-1,J),Sfc(I-1,J+1),Sfc(I,J-1),Sfc(I,J+1),Sfc
(I+1,J-1),Sfc(I+1,J),Sfc(I+1,J+1))
2235      IF Point>Max OR Point<Min THEN
2240       CALL Label(1,.6,0,5,4,(J),(I),"+")
2245       IF Point>Max THEN
2250        CALL Label(3,.6,0,5,4,(J),(I),"H")
2255       ELSE
2260        CALL Label(3,.6,0,5,4,(J),(I),"L")
2265       END IF
2270       IF Extremes>1 THEN
2275        CALL Label(3,.6,0,5,4,J,I+.2,"")        ! No label, just setup
2280        LABEL USING Image$;Point
2285       END IF
2290      END IF
2295     NEXT J
2300    NEXT I
2305   END IF ! (if Stats)
2310   MOVE X_gdu_max,Y_gdu_max
2315   PEN 0
2320  SUBEND
2325  !
2330  ! ------------------------------------------------------------------
2335  !
2340 Show:SUB Show(Xleft,Xright,Ylow,Yhigh)
2345     !  This simulates the system command SHOW, but saves the variables so
2350     !  the routines Setgu and setuu work
2355   COM /G_units/ Gdu_xmax,Gdu_ymax,Udu_xmin,Udu_xmax,Udu_ymin,Udu_ymax,Show
2360   IF Gdu_xmax=0 THEN
2365    Gdu_xmax=100*MAX(1,RATIO)
2370    Gdu_ymax=100*MAX(1,1/RATIO)
2375   END IF
2380   Udu_xmin=Xleft
2385   Udu_xmax=Xright
2390   Udu_ymin=Ylow
2395   Udu_ymax=Yhigh
2400   Show=1
2405   SHOW Xleft,Xright,Ylow,Yhigh
2410  SUBEND
2415 Gdu:SUB Gdu(X_gdu_max,Y_gdu_max,OPTIONAL Gdu_xmid,Gdu_ymid)
2420   ! This returns Xright, Yhigh and their respective midpoints in GDUs.
2425   ! Note that if Gud_xmid is defined, Gdu_ymid must be also.
2430   COM /G_units/ Gdu_xmax,Gdu_ymax,Udu_xmin,Udu_xmax,Udu_ymin,Udu_ymax,Show
2435   IF Gdu_xmax=0 THEN
2440    Gdu_xmax=100*MAX(1,RATIO)
2445    Gdu_ymax=100*MAX(1,1/RATIO)
2450   END IF
2455   X_gdu_max=Gdu_xmax
2460   Y_gdu_max=Gdu_ymax
```

```
2465   IF NPAR>2 THEN
2470    Gdu_xmid=Gdu_xmax*.5
2475    Gdu_ymid=Gdu_ymax*.5
2480   END IF
2485  SUBEND
2490  !
2495  !
2500  !
2505 Label:SUB Label(Csize,Asp_ratio,Ldir,Lorg,Pen,X,Y,Text$)
2510 ! This defines several systems variables (in CSIZE, LDIR, etc.), and
2515 ! labels the test (if any) accordingly.
2520   DEG
2525   CSIZE Csize,Asp_ratio
2530   LDIR Ldir
2535   LORG Lorg
2540   PEN Pen
2545   IF Text$<>"" THEN
2550    MOVE X,Y
2555    LABEL USING "#,K";Text$
2560   END IF
2565   PENUP
2570  SUBEND
2575  !
2580  !
2585  !
2590 Scale:SUB Scale(Surface(*),New_min,New_max)
2595   OPTION BASE 1
2600  ! This routine scales a matrix such that it will have a new lowest
2605  ! value of New_min and a new higest value of New_max.
2610   DISP USING "K";"Scaling the surface array from ",New_min," to ",New_max,"
."
2615   Min=MIN(Surface(*))
2620   Max=MAX(Surface(*))
2625   IF Min=Max THEN ! Array is completely flat
2630    MAT Surface= (New_min)
2635    SUBEXIT
2640   END IF
2645   MAT Surface= Surface-(Min)
2650   Range_recip=(New_max-New_min)/(Max-Min)
2655   MAT Surface= Surface*(Range_recip)
2660   MAT Surface= Surface+(New_min)
2665   DISP
2670  SUBEND
```